Sunhayato

# R8C/Tiny マイコンシリーズ

## スタートアップガイド

### E8エミュレータ導入編


2006年4月10日発行日

REV. 2. 00

SG043101


サンハヤト株式会社

本社 〒170-0005 東京都豊島区南大塚3-40-1

☎ 03-3984-7791　FAX. 03-3971-0535

http://www.sunhayato.co.jp

# 安全上のご注意

このたびは、弊社製品をご使用いただき、誠にありがとうございます。本項では、誤った取り扱いによる事故を未然に防ぐための安全上の注意事項を説明しています。弊社製品をご使用になる前に必ずお読みください。

| | |
|---|---|
| ⚠警告 | この表記を無視して誤った取り扱いをすると、死亡や重傷など、人体への重大な障害をもたらす恐れのある内容について示しています。 |
| 注意 | この表記を無視して誤った取り扱いをすると、軽傷または中程度の障害をもたらす恐れのある内容について示しています。また、本品や本品に接続している機器に損傷を与える可能性がある事項についても示しています。 |

## 警告

| | |
|---|---|
| <br>プラグをコンセントから抜く | ▶**発煙、異臭への対処**<br>煙がでている、へんな臭いがするなどの異常がありましたら使用を直ちに中止してください。そのまま使用すると、火災、感電、故障の原因となります。<br>すぐに電源ケーブルのプラグをコンセントから抜き、煙などの異常が出なくなるのを確認し、販売店などに修理をご依頼ください。 |
| <br>水場禁止 | ▶**水分の多いところ、水がかかる場所では、本製品は使用しないで下さい**<br>風呂場や台所など水分の多いところ、水がかかる場所では、本製品は使用しないで下さい。 火災、感電、故障の原因となります。 |
| <br>禁止 | ▶**電源ケーブルの取り扱いに注意してください**<br>電源ケーブルを傷つけ、破損、加工、無理な曲げ、引っ張り、ねじり、束ねたりしないでください。また、重い物を乗せたり、加熱したりすると電源ケーブルが破損し、火災、感電、故障の原因となります。 |
| <br>禁止 | ▶**医療、軍事、航空宇宙、列車、運送、原子力などの制御設備へは使用しないでください**<br>医療機器、軍事機器、航空宇宙機器、運送、原子力などの制御設備などの人命に関わるシステムへの使用は意図しておりません。 |
| <br>接触禁止 | ▶**雷が鳴りはじめたら製品に触れないでください**<br>近くに雷が発生したときは、製品本体に触れないでください。また、電源プラグをコンセントから抜いてご使用をお控えください。雷によっては、火災、感電、故障の原因となることがあります。 |

## 注意

| | |
|---|---|
| <br>電源プラグの差し込み | ▶**電源プラグは確実にコンセントに差し込んでください**<br>差し込みが不完全ですと火災、感電、過熱の原因になります。 |
| <br>分解禁止 | ▶**分解・改造しないでください**<br>分解、改造しないでください。怪我、感電、故障の原因となります。本製品の分解、改造による怪我や事故について、当社は責任を負いかねます。 |
| <br>接触禁止 | ▶**濡れた手での操作は避けてください**<br>濡れた手で電源ケーブル・プラグを抜き差ししないでください。また、製品に触れないでください。感電の原因となることがあります。 |

# 注意

| | |
|---|---|
| 禁止 | ▶以下のような場所では使用しないでください<br>本製品を以下のような場所で使用すると、動作不良、故障の原因となります。<br>・振動や衝撃が加わる場所<br>・直射日光のあたる場所<br>・湿気やホコリが多い場所<br>・温度差の激しい場所<br>・熱を発生するもの（暖房器具など）の近く<br>・強い磁力、電波が発生するもの（磁石、ディスプレイ、スピーカー、ラジオ、無線機など）の近く<br>・湿気の多い場所 |
| <br>子供注意 | ▶子供の手の届かない場所に置いてください<br>本製品に装着されている電子部品など子供が飲み込まないように注意してください。 |
| <br>活線挿抜禁止 | ▶通信ケーブルの抜き差しは、電源ＯＦＦにして行ってください<br>本品への通信ケーブル類は活線挿抜しないでください。ケーブルの抜き差しは、必ず本製品または相手製品の電源がＯＦＦ状態にて行ってください。故障の原因になることがあります。 |
| <br>安全設計 | ▶安全設計をしてください<br>本製品を、高度な信頼性を必要とするシステムに使用する場合は、冗長設計、誤動作防止設計など充分な安全設計を必ず行ってください。本製品の故障、傷害により生じるいかなる損害、事故について当社は責任を負いかねます。 |
| <br>保管注意 | ▶長期間使用しない場合の保管について<br>長期間使用しない場合は、帯電防止袋などに入れ、ホコリなどが入らないようにしてください。ホコリが入ると接触不良などの原因になります。 |
| <br>ホコリ注意 | ▶製品の清掃について<br>製品にホコリなどが付着すると放熱特性が低下し、故障の原因になりますので、下記の「▶お手入れについて」に従って清掃してください。 |
| <br>薬品注意 | ▶お手入れについて<br>汚れはやわらかい布によるからぶきをしてください。水、洗剤、ベンジン、シンナーなどの使用は避けてください。基板の洗浄には、サンハヤト製電子機器用洗浄剤をお使いください。 |
| <br>使用注意 | ▶故障、破損時の処理について<br>本製品が故障もしくは破損した場合は、速やかに使用を中止して販売店などに修理依頼してください。そのまま使用しますと火災、感電、怪我の原因になるおそれがあります。 |
| <br>廃棄注意 | ▶本製品の廃棄について<br>本製品の廃棄は、各自治体の廃棄ルールに従ってください。詳しくは各自治体にお問い合わせください。 |
| <br>AC100V<br>以外禁止 | ▶日本国内のみで使用してください<br>本製品は日本国内の商用 AC100V 電源仕様です。海外では使用できません。AC100V 以外では絶対に使用しないでください。 |

# 本資料についてのご注意

## 本資料について

- 本資料は、電子工作や電子回路、パーソナルコンピュータの操作について一般的な知識をお持ちの方を対象にしています。
- 本資料を元に操作するには、株式会社 ルネサス テクノロジ社製 R8C/Tiny マイコンについての知識や開発環境などが必要です。
- Microsoft®、Windows® は米国 Microsoft 社の米国およびその他の国における商標登録です。
- その他、記載されている製品名は各社の商標または商標登録です。

## 本資料のご利用にあたって

- この取扱説明書に掲載している内容は、お客様が用途に応じた適切な製品をご購入頂くことを目的としています。その使用により当社及び第三者の知的財産権その他の権利に対する保証、又は実施権の許諾を意味するものではありません。また、権利の侵害に関して当社は責任を負いません。
- 本資料に記載した情報を流用する場合は、お客様のシステム全体で充分評価し適用可能かご判断願います。当社では適用可能判断についての責任は負いません。
- 本資料に記載してある内容は、一般的な電子機器（学習教材、事務機器、計測機器、パーソナル機器、コンピュータ機器など）に使用されることを目的としています。高い品質や信頼性が要求され、故障や誤作動が直接人命を脅かしたり人体に危害を及ぼす恐れのある、医療、軍事、航空宇宙、原子力制御、運輸、移動体、各種安全装置などの機器への使用は意図も保証もしておりません。
- この取扱説明書の一部、又は全部を当社の承諾なしで、いかなる形でも転載又は複製されることは堅くお断りします。
- 全ての情報は本資料発行時点のものであり、当社は予告なしに本資料に記載した内容を変更することがあります。
- この資料の内容は慎重に制作しておりますが、万一記述誤りによってお客様に損害が生じても当社はその責任を負いません。
- 本資料に関してのお問合せ、その他お気付きの点がございましたら、当社までお問合せください。
- 本資料に関する最新の情報はサンハヤト株式会社ホームページ（http://www.sunhayato.co.jp/）に掲載しております。

# このマニュアルについて

本スタートアップガイドでは、サンハヤト R8C/Tiny マイコンシリーズのプログラム開発を、ルネサス テクノロジ社の E8 エミュレータおよび HEW（High-Performance Embedded Workshop）を使用して行った場合の手順、設定方法などを説明したものです。E8 エミュレータ、HEW の詳細な使用方法や注意事項につきましては、ルネサス テクノロジ社発行のマニュアルを参照してください。

## ■該当するサンハヤトR8C/Tiny マイコンシリーズ製品

● ターゲットマイコン ：SR8C15CP、MB-R8CS（R5F21154SP 搭載マイコンボード）
● 書き込みボード ：MB-RS8、MB-R8C-SET、CT-208

## ■E8エミュレータを使用した開発に必要なもの

● ホストパーソナルコンピュータ（以下の条件を満たすもの）

| PC 本体 | Pentium Ⅲ 600MHz 以上を搭載した IBM PC/AT 互換機 |
|---|---|
| OS | Windows XP, Windows Me, Windows 98SE, Windows 2000（Windows XP 推奨） |
| メモリ | 128MB 以上（ロードモジュールの 2 倍以上） |
| ハードディスク | エミュレータ用ソフトウェアのインストールには 100MB 以上の空き容量が必要 |
| ディスク装置 | CD-ROM ドライブ（インストール用） |
| 入力デバイス | マウス、またはマウス相当のポインティングデバイス |
| インターフェイス | USB（USB 1.1、フルスピード） USB2.0 対応のホストコンピュータにも接続できます。 |

● サンハヤト R8C/Tiny マイコンシリーズ製品（ターゲットマイコン＋書き込みボード）

## ■このマニュアルで使用しているツール類

このスタートアップガイドでは、以下の OS、ツールのバージョンで動作したものとして説明しています。
● ホスト PC の OS ：Windows XP Professional
● E8 エミュレータファームウェア ：1.07.02.000
● 統合化開発環境 ：HEW V.4.00.03.001
● C コンパイラ ：NC30WA V.5.40 Release 00A
E8 エミュレータ、HEW のバージョンは、E8 エミュレータ同梱の CD の Je8rm.txt で確認できます。また HEW のバージョンは、HEW の「ヘルプ」メニューの「High-Performance Embedded Workshop のバージョン情報」でも確認できます。

## ■このマニュアルで紹介している各社サイト、ツールについて

本スタートアップガイドで紹介している各社サイトやツールは、本スタートアップガイド発行時のものを掲載しております。各社サイトの構成やツールのバージョン、またバージョンに伴ってツールのファイル名などが変わっている場合がありますのでご了承ください。

# 目次

# コラム目次

# 1. E8 エミュレータによる開発

## 1.1 E8 エミュレータについて

E8 エミュレータはルネサス テクノロジ社 R8C/Tiny シリーズ マイクロコンピュータ専用のオンチップデバッギングエミュレータです。以下に特長を示します。

■ ホスト PC との接続

ホスト PC とは USB ケーブルで接続します（USB1.1 または USB2.0）。ホスト PC から E8 エミュレータへの電源供給はもちろん、E8 エミュレータを介してターゲットボードへ電源を供給することもできます。ターゲットへの電源供給は 3.3V と 5.0V から選択でき、最大 300mA の出力が可能です

■ ターゲットシステムとの接続

ターゲットマイコンとは MODE 端子（ユーザーシステムでは使用しない端子です）と RESET 端子で通信します。

■ 統合開発環境

E8 エミュレータを制御するホスト PC 側のアプリケーションとしては統合開発環境 HEW（High-performance Embedded Workshop）を使用することになります。

■ E8 エミュレータで評価時の注意事項

E8 エミュレータはターゲットマイコンに書き込んだファームウェアと通信し、マイコン内部の情報をホスト PC に送ります。ターゲットマイコンはファームウェアとユーザープログラムを交互に実行することになります。この原理をふまえ、注意事項に留意しておくことで HEW および E8 エミュレータを使用した開発環境を最大限に活用することができます。

評価時のファームウェアによるマイコン内部資源の制限事項についてはルネサス テクノロジ社発行の「E8 エミュレータ　ユーザーズマニュアル別冊（R8C/14, R8C/15, R8C/16, R8C/17 接続時の注意事項）」（rjj10j0726_e8_s.pdf）の「E8 エミュレータと各 MCU の相違点」の項を参照してください。

以下にユーザープログラム評価時の SR8C15CP マイコンのメモリマップを示します。

■ ファームウェア領域

E8 エミュレータ起動時に、ターゲットマイコンにファームウェアが書き込まれます。書き込む領域は、データ領域（2400h ～ 2BFFh）か、プログラム領域（C000h ～ C7FFh）かを選択できます。

■ スタックエリア

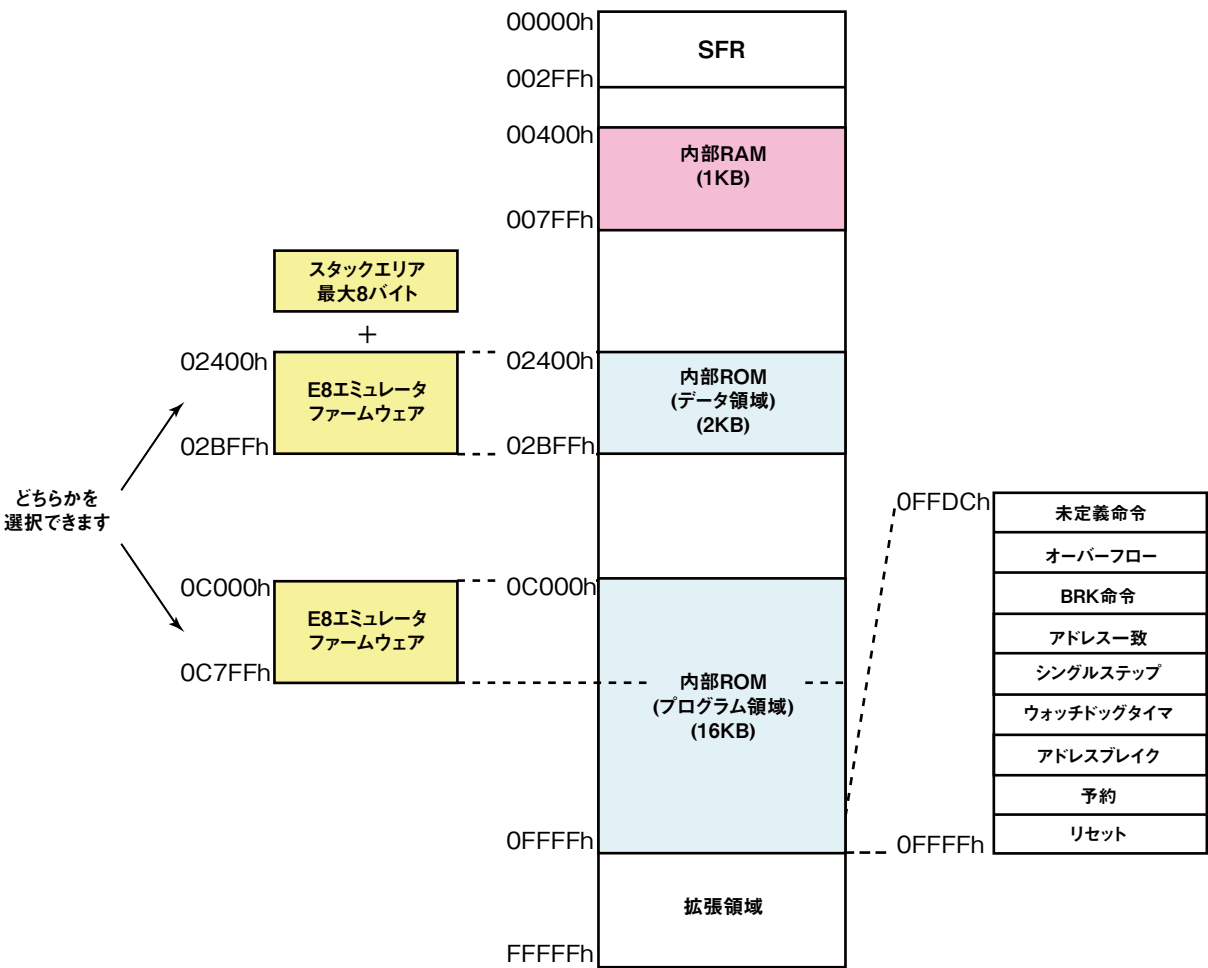
ユーザープログラムブレーク時にファームウェアが使用するスタックエリアを最大 8 バイトを考慮する必要があります。スタックエリアには 8 バイト分の余裕を確保してください。

SFR: Special Function Register の略で、主に周辺機能のレジスタが配置されています。
内部 ROM（データ領域）: Flash ROM で、書き込み / 消去が 10,000 回可能です。
内部 ROM（プログラム領域）: Flash ROM で、書き込み / 消去が 100 回可能です。

**図1-1 E8エミュレータで評価時のSR8C15CPマイコン メモリマップ**

## 1.2 開発の流れ

HEW は、プログラムの作成 / 編集から、E8 エミュレータで動作させるための最終ゴールファイルの生成、E8 エミュレータでの実機デバッグ制御、最終リリース時のユーザープログラム書き込み（ファームウェア消去）までの各フェーズを同一インターフェースで行える統合開発環境です

E8 エミュレータを使用した HEW 開発フロー（イメージ図）を以下に示します。

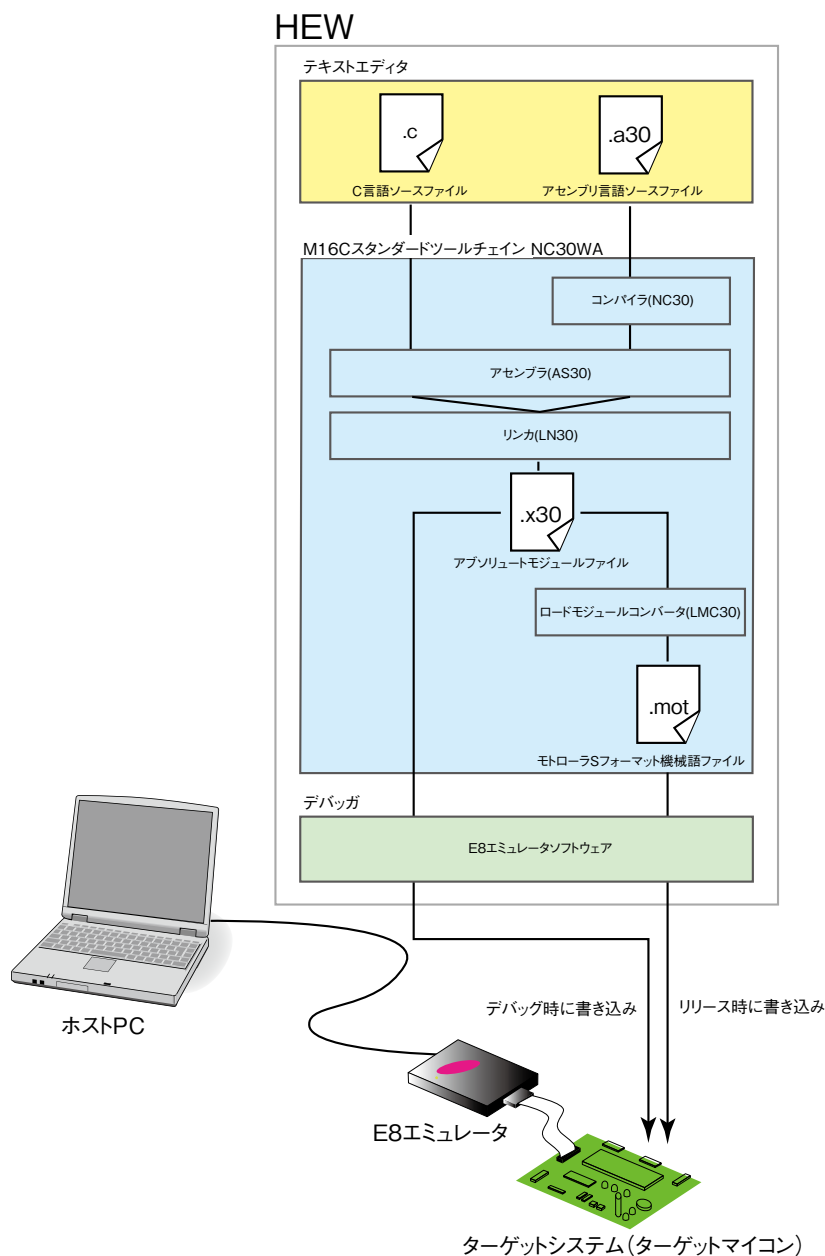



**図1-2 HEW と E8 エミュレータによる開発フロー（イメージ図）**

# 2. 開発環境を整えよう

## 2.1 E8 エミュレータによる開発でインストールするもの

### ■E8エミュレータソフトウェア

「E8 エミュレータソフトウェア」には HEW、コンパイラ（NC30WA)、E8 エミュレータデバッガソフトウェアが含まれています。E8 エミュレータ同梱の CD に収められています。

最新バージョンのものがルネサステクノロジ社サイト (http://japan.renesas.com/homepage.jsp) からもダウンロードできます。(トップページ > マイコン > R8C/Tiny> 開発環境 > デバッガ / エミュレータ >E8> ダウンロードより、「E8 エミュレータ」を選んでください。(アップデータでも初期インストールに使用できます。)

### ■E8エミュレータ専用USBドライバソフトウェア

「E8 エミュレータ専用 USB ドライバソフトウェア」は、E8 エミュレータとホスト PC を USB 接続するためのドライバソフトウェアです。E8 エミュレータ同梱の CD に収められています。

## 2.2 インストールの手順

E8 エミュレータ同梱の CD からインストールする場合は、CD 内の「setup.exe」を、ルネサステクノロジ社サイトより「E8 エミュレータソフトウェア」をダウンロードした場合は、「e8fullv206r02.exe」（バージョン、リリースによってファイル名は異なります)をダブルクリックしてください。インストールの際に「セキュリティの警告」ウィンドウが出ますが、「実行」を押してください。

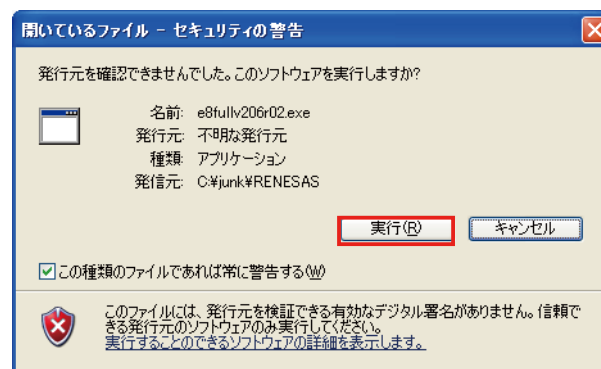
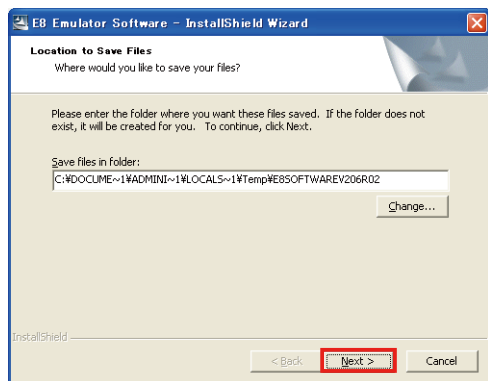
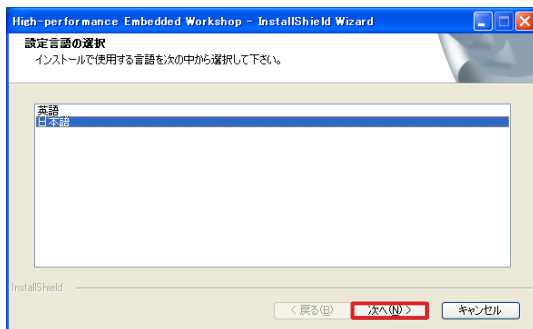
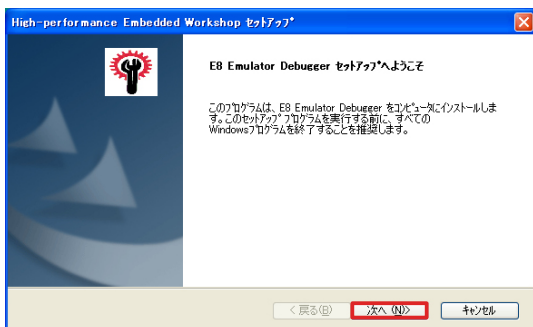
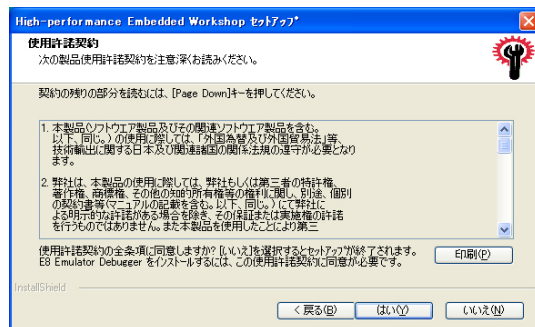
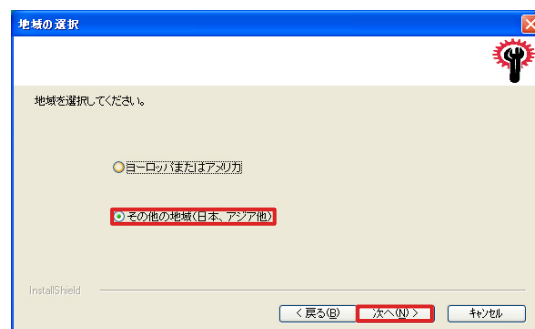



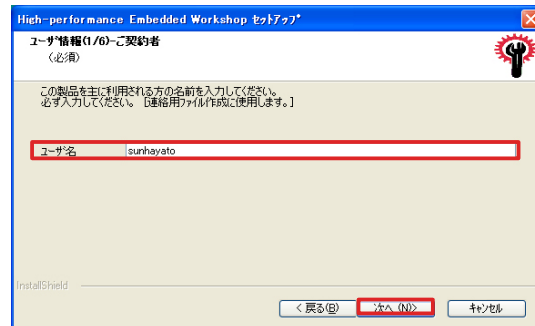
**図2-1 セキュリティの警告**

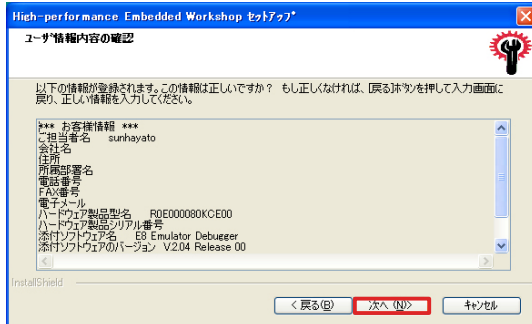
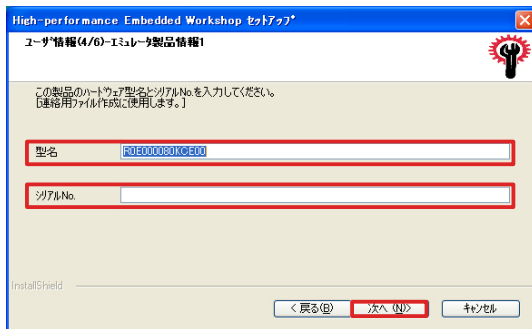
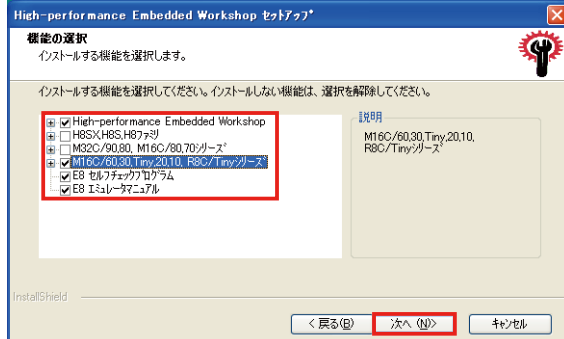
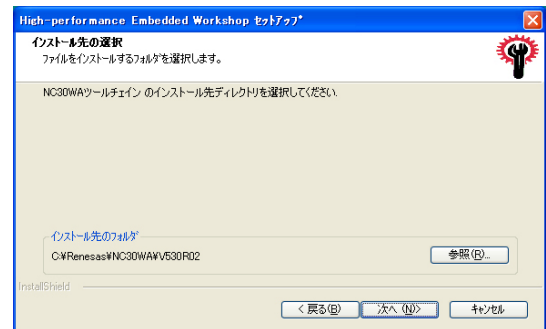
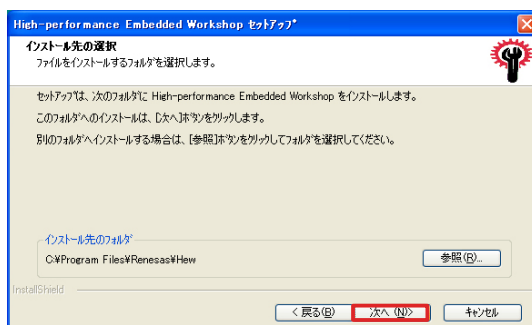
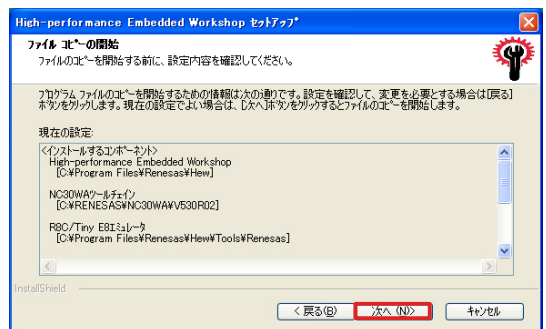
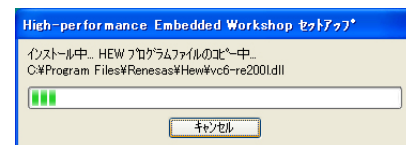
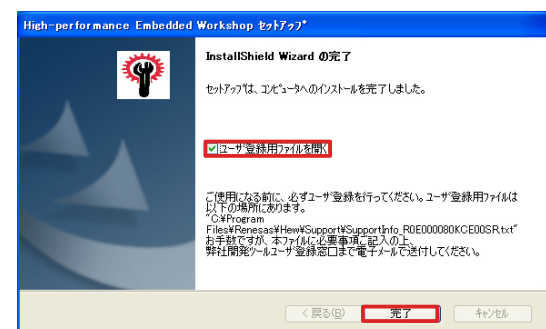
## ❶ E8 エミュレータ S/W のインストール

解凍のためのテンポラリフォルダの指定画面が開きます。そのまま「次へ」をクリックしてください。

インストールするソフトウェアは、「E8 エミュレータソフトウェア」、「オートアップデートユーティリティ」の両方にチェックが入った状態で「次へ」をクリックしてください。

設定言語の選択では、日本語が選択されている状態で「次へ」をクリックしてください。

「E8 Emulator Debugger セットアップへようこそ」画面で「次へ」をクリックしてください。セットアップが開始されます。

「使用許諾契約」の内容を確認したら、「次へ」をクリックしてください。

「地域の選択」では「その他の地域（日本、アジア他）」が選択されている状態で「次へ」をクリックしてください。

「ユーザ情報」は、契約者名のみ必須です。ここで入力した情報は、連絡ファイル（E8 エミュレータ製造元のルネサステクノロジ社のサポートを受ける場合に使用すると便利なテキストファイル）の項目として使用されます。

「個人情報保護方針」を確認して「OK」をクリックしてください。

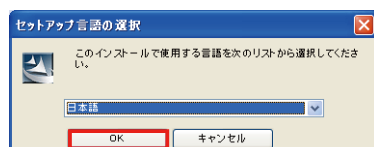
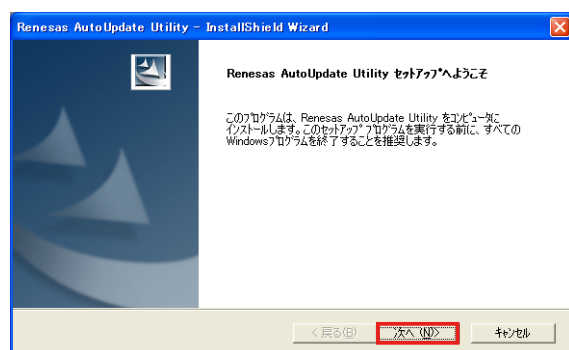
「ユーザー情報 2/6 ～ 5/6」まで必要に応じて情報を入力してください。

シリアル No. は E8 エミュレータ本体裏に記載されています。

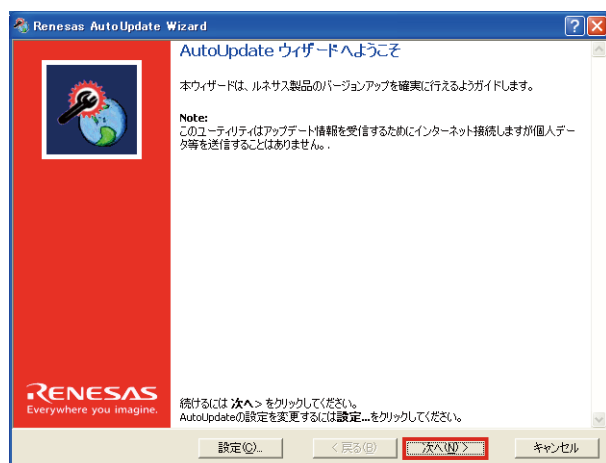
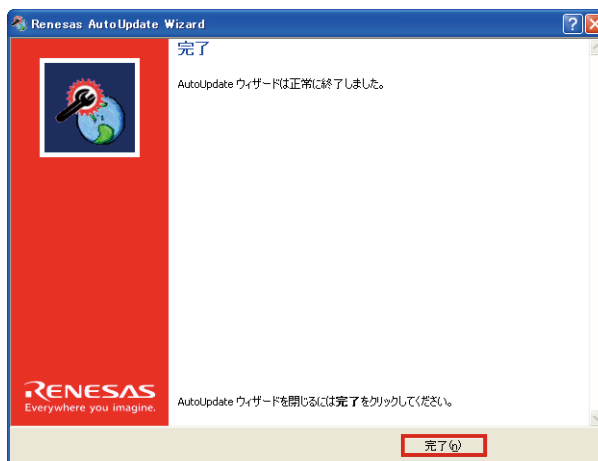
入力した情報を確認したら「次へ」をクリックしてください。

「インストール機能の選択」では、サンハヤトR8C/Tinyシリーズをお使いの場合は、「High-performance Embededd Workshop」、「M16C/60,30,Tiny,20,10,R8C/Tinyシリーズ」にチェックしてください。あとは、必要に応じて機能を選択してください。

HEW、NC30WAツールチェインのインストール先は、特に問題がなければデフォルトのままにしておくことをお勧めします。「次へ」をクリックしてください。

「ファイルコピーの開始」ではコピー先を確認して「次へ」をクリックしてください。

ファイルがコピーされます。

「連絡用ファイルを開く」にチェックを入れて、「完了」をクリックすると、入力したユーザー情報が入ったテキストファイルが開きます。ユーザー登録にご利用ください。

## 2 AutoUpdate の設定

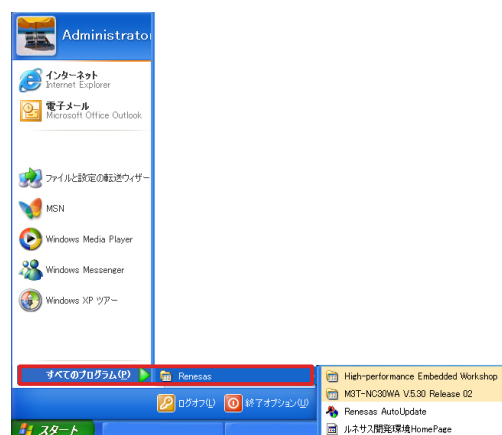
引き続き、AutpUpdate の設定を行います。

「AutoUpdate」とは、インターネットに接続している環境において、設定したタイミングで自動的にルネサステクノロジ社のサイトへアクセスし、ツールのアップデートがあった場合にダウンロード、アップデートを行うものです。

(「AutoUpdate」を設定すると、HEW が起動していないときでも、アップデータが Windows に常駐します。)

「セットアップ言語の選択」では、「日本語」が選択されている状態で「OK」をクリックしてください。

「AutoUpdate」のインストールウィザードが開始します。「次へ」をクリックしてください。

インストールが完了したら「OK」をクリックしてください。

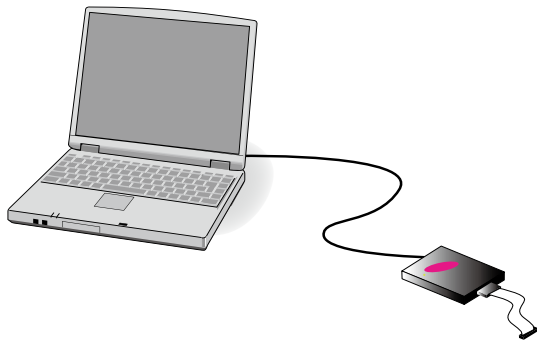
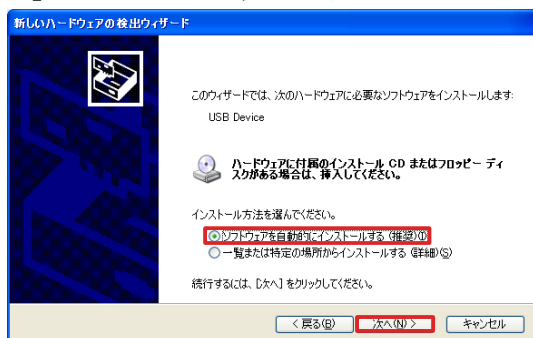
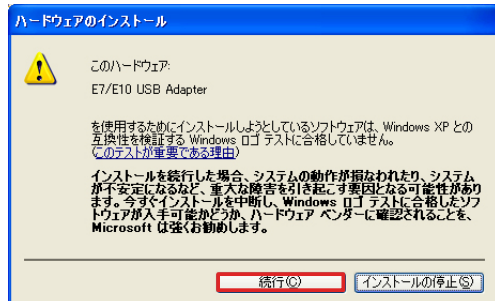
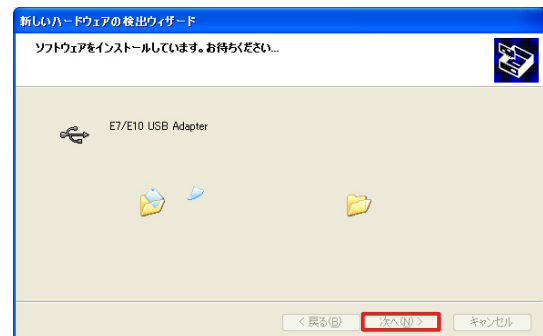
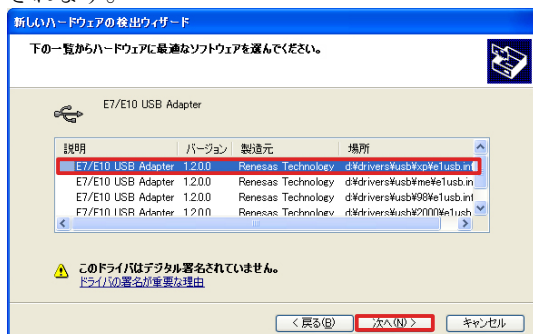
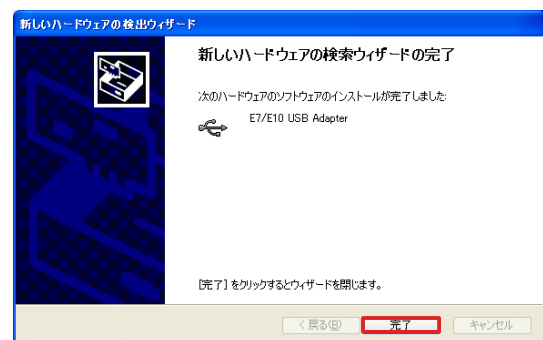
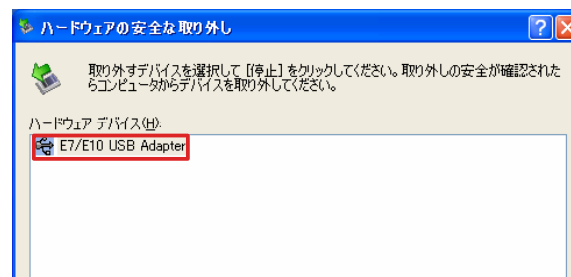
Windows 画面のタスクトレイに表示されている「AutoUpdate」アイコンが「アップデートあり」の表示 になりますので、クリックしてウィザードに従って設定してください。

ウィザードが完了したら、「完了」をクリックしてください。

## 3 インストールの確認

Windows の「スタート」→「すべてのプログラム」より「Renesas」各ツールがインストールされてることが確認できます

## 4 USB ドライバのインストール

E8 エミュレータとホスト PC 間を USB ケーブルで接続し、USB ドライバをインストールします。

(E8 の初回の接続時のみ行います。)

E8 エミュレータ用 USB ドライバは、E8 エミュレータ同梱の CD に収められています。ホスト PC の OS が Windows XP の場合は、管理者または Administrators グループのメンバとしてログオンしている必要があります。

E8 エミュレータとホスト PC を接続してください。

「新しいハードウェアの検索ウィザードの開始」ウィンドウが開きます。「ソフトウェアを自動的にインストールする（推奨）」にチェックが入った状態で「次へ」をクリックしてください。

「Windows ロゴテスト」についての警告が出ますが、「続行」をクリックしてください。

E8 エミュレータ付属の CD より、USB ドライバが表示されます。該当する OS のドライバを選択して「次へ」をクリックしてください。ここでは Windows XP にインストールすることとして、CD-ROM ドライブ：¥drivers ¥usb ¥xp ¥e1usb.sys を選択します。USB ドライバのインストールが開始されます。

USB ドライバのインストールが完了します。「完了」をクリックしてください。

ホスト PC から E8 エミュレータを外す場合は、必ず Windows の「ハードウェアの安全な取り外し」インジケータをクリックし、「E7/E10USB Adapter」の「停止」を実行して E8 エミュレータのパワーインジケータをが消灯しているのを確認してから行ってください。

# 3. サンプルプログラムを動かそう

## 3.1 サンプルプログラムのダウンロード

当社 Web サイトの R8C/Tiny マイコンシリーズのページ (http://www.sunhayato.co.jp/tiny/) よりサンプルプログラム（LED_PICOPICO.lzh）をダウンロードしてください。サンプルプログラムは HEW で開発したものです。Lzh 形式の圧縮ファイルですので、解凍する必要があります。

以降紹介する例では、コピーした LED_PICOPICO.lzh を C:¥WorkSpace フォルダへ解凍したこととしています。

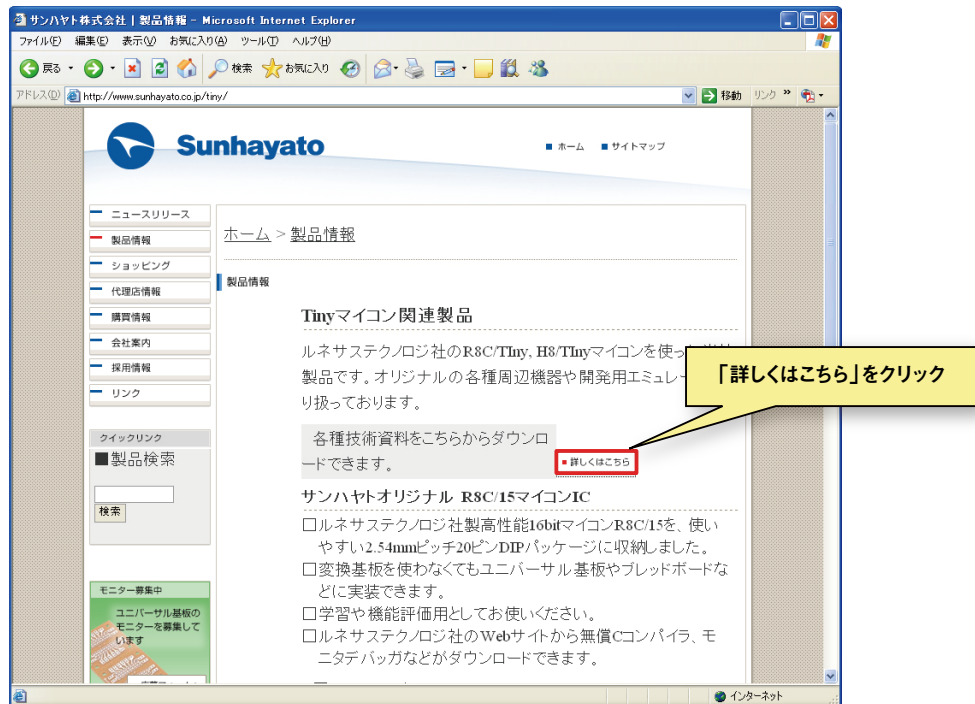



図 3-1 R8C/Tinyマイコンシリーズのページ

# 3.2 サンプルプログラムについて

## ■サンプルプログラムLED_PICOPICOの動作

LED_PICOPICO を動作させせると、ターゲットボード上の 2 つの LED が 0.1 秒ごとに点滅します。
各 LED はそれぞれ、ポート 1_1（p1_1）、ポート 1_2（p1_2）に接続されているものとします。

**表3-1 各ボードとLEDの対応表**

| サンハヤト R8C/Tiny シリーズ | 点滅する LED | 接続されているポート |
|---|---|---|
| MB-RS8 | LED4（緑） | p1_1 |
| | LED5（橙） | p1_2 |
| CT-208 | LED1 | p1_2 |
| | LED2 | p1_1 |

## ■ サンプルプログラムLED_PICOPICOの構成

HEW では、プログラムのソースファイルと、ソースファイルからオブジェクトファイル（マイクロコンピュータが実行可能な機械語ファイル）を生成するまでの手順や設定をまとめて、1 つの「プロジェクト」として扱います。

また､「プロジェクト」はさらに「ワークスペース」という大きなグループに属しています。1 つのワークスペースに複数のプロジェクトが属している場合もあります。

サンプルプログラムでは LED_PICOPICO というワークスペースに同名のプロジェクトが 1 つ属しています。以下にサンプルプログラムの構成を示します。

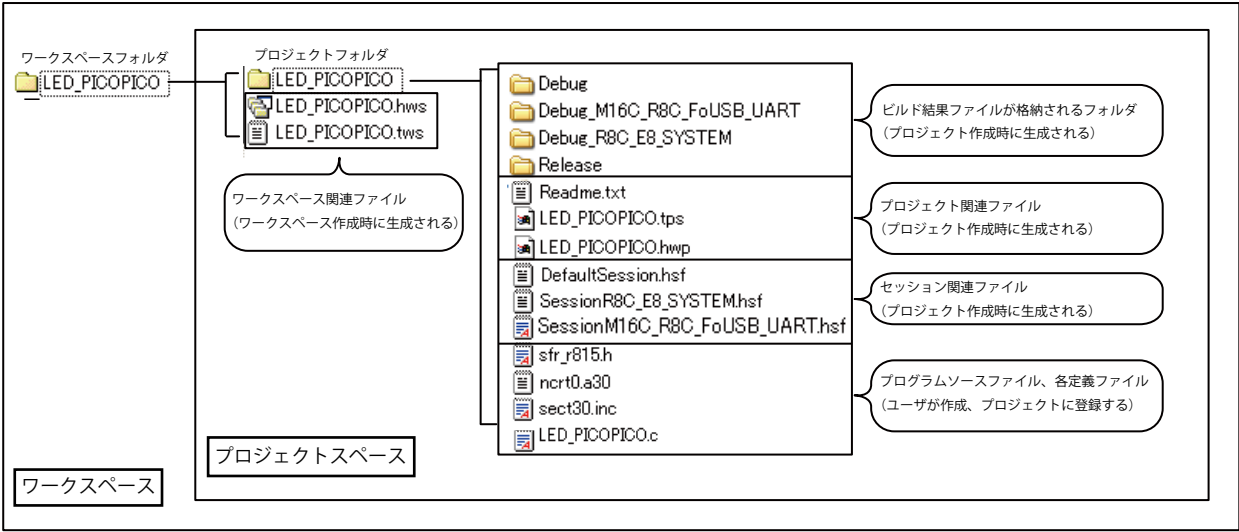



**図 3-2 サンプルプログラムの構成**

## ■ 各設定ファイルの内容

ソースファイル群の各設定ファイルは、コンパイラ（アセンブラ）NC30WA 標準のファイル（一部修正を加えています）を使用しています。各ファイルの概要を以下に示します。

**表3-2 サンプルプログラムの各設定ファイル**

| ファイル名 | 内容 | 変更箇所 |
|---|---|---|
| ncrt0.a30 | スタートアップルーチン | ・heap 領域の初期化箇所をコメントアウト<br>・標準入出力の初期化箇所をコメントアウト |
| sect30.inc | セクション定義ファイル | プログラムの開始アドレスを E000h から C000h に変更 |
| sfr_r815.h | R8C/15 シリーズ SFR 定義ヘッダファイル | - |

スタートアップルーチン ncrt0.a30 は SR8C15CP マイコンのリセット後に実行されます。以下に処理内容を示します。

・割り込みスタックポインタ（ISP）にベースアドレスを設定する（領域サイズは 80h）
・プロセッサモードレジスタ 0 に 00h を設定する（ソフトウェアリセットビットが 0）
・フラグレジスタに 80h を設定する（割り込み許可）
・ユーザスタックポインタ（USP）にベースアドレスを設定する（領域サイズは 80h）
・スタティックベースレジスタ（SB）にベースアドレス 400h を設定する
・割り込みテーブルレジスタ（可変ベクタテーブルの先頭アドレス）に 0000FEDCh を設定する
・初期値のない near、far データ領域をゼロクリアする
・初期値のある near、far データ領域に、初期値が格納された領域から初期値をコピーする
・main 関数を呼び出す。

## ■ メイン関数の内容

以下に main.c のリストを示します。タイマ X の割り込み要求フラグをポーリングで検知し、カウントすることで 1 秒間隔の LED 点滅を実現しています。

```
/*------------------------------------------------------------------------
 * FILE              :LED_PICOPICO.c
 * DATE              :2006/3/20
 * DESCRIPTION       :Main Program
 * CPU TYPE          :R8C/Tiny 15Group (20MHz)
 * This file is generated by Renesas Project Generator(Ver.4.00.03.001)
 *------------------------------------------------------------------------*/
/*------------------------------------------------------------------------
 *
 *SR8C15CP　サンプルプログラム（0.5秒間隔LED点滅 タイマXによる計測）
 *
 *              サンハヤト株式会社
 *------------------------------------------------------------------------*/
/* インクルードファイル */
#include "sfr_r815.h"                          /* SR8C15CPのSFRレジスタ定義ヘッダ（C言語用）*/
/* プロトタイプ宣言 */
void init(void);                               /* システムクロック、SFR(Special Function Register)の初期設定*/
void led_blink_out(void);                      /* ポートの状態を反転して出力する関数 */
/*------------------------------------------------------------------------
 * ID                :なし
 * 関数名            :main
 * 機能              :2つのLEDを点滅させる（MB-RS8→p1_1：LED4、p1_2：LED5　CT-208→p1_1：LED2、p1_2：LED1）
 * 引数              :なし
 * 戻り値            :なし
 * 使用関数          :init(),led_blink_out()
 *------------------------------------------------------------------------*/
void main(void)
{
        unsigned int timerX_count_500ms=0;     /* 500ms計測用カウンタ */
        init();     /* システムクロック、SFRの初期設定 */
        while(1)                               /* 無限ループ */
        {
                if(ir_txic==1)                 /* タイマX割り込み要求があれば20msec経過 */
                {
                        timerX_count_500ms++;
                        if(timerX_count_500ms >= 25)   /* 500msカウント（20msec×25=0.5sec） */
                        {
                                led_blink_out();       /* ポートレジスタの値を反転させることでLEDが点滅する*/
                                timerX_count_500ms = 0; /* 500ms計測用カウンタ初期化 */
                        }
                        txic=txic&0b00000111;  /* タイマx割り込み要求フラグ(ir_txic=3ビット目)はMOV命令でクリアする */
                }
        }
}
```

**リスト3-1 LED_PICOPICO メイン関数**

```
/*--------------------------------------------------------------------------
 * ID                    : なし
 * 関数名                : init
 * 機能                  : システムクロックにメインクロック（外付けセラミック発振子20MHz）を設定、SFRの初期設定
 * 引数                  : なし
 * 戻り値                : なし
 * 使用関数              : なし
 *--------------------------------------------------------------------------*/
void init(void)
{
        /* システムクロック初期設定 */
        asm("FCLR I");                  /* 割り込み禁止 */
        prc0 = 1;                       /* CM0,CM1,OCDレジスタ プロテクト解除 */
         cm13 = 1;                      /* Xin Xout端子に設定 */
        cm15 = 1;                       /* Xin-Xout 駆動能力: HIGH */
        cm05 = 0;                       /* メインクロック発振 */
        cm16 = 0;                       /* CM16,CM7=00 →システムクロック分周なし */
        cm17 = 0;
        cm06 = 0;                       /* システムクロック分周比→CM16,CM17有効 */
        asm("nop");                     /* 発振安定までウェイト */
        asm("nop");
        asm("nop");
        asm("nop");
        ocd2 = 0;                       /* システムクロックにメインクロックを選択 */
        prc0 = 0;                       /* CM1,CM2,OCDレジスタ プロテクト設定 */
        asm("FSET I");                  /* 割り込み許可 */

        /* 入出力ポート初期設定 LED1=p1_2、LED2=p1_1 */
        drr2 = 1;                       /* p1駆動能力制御レジスタdrr2(LED5or1)に1を設定する */
        drr1 = 1;                       /* p1駆動能力制御レジスタdrr1(LED4or2)に1を設定する */
                                        /* 1に設定すると1ピンあたり約15ｍAの電流を引き込める */

        p1 = 0b00000010;                /* p1レジスタにLEDの初期状態（p1_2(LED5or1)→0:点灯、p1_1(LED4or2)→1:消灯）を設定する */
                                        /* この時点ではまだ入力ポートの設定になっているのでリードモディファイライト命令は使用しない */

        pd1_2 = 1;                      /* p1方向レジスタp1_2(LED5or1)を出力に設定する */
        pd1_1 = 1;                      /* p1方向レジスタp1_1(LED4or2)を出力に設定する */

        /* タイマX初期設定 */
        txic = 0b00000000;              /* タイマX割り込み制御レジスタ→タイマX割り込み禁止 */
        txmr = 0b00000000;              /* タイマXモードレジスタ→タイマモード、カウント停止 */
        tcss = 0b00000001;              /* タイマXカウントリソース設定レジスタ→f8(20MHz/8=2.5MHz) */
        prex = 250-1;                   /* タイマXレジスタ、プリスケーラにカウント値設定 */
        tx = 200-1;                     /* 0.4μsec*50000=20msec　5000を250*200に分けて設定する */
        txs = 1;                        /* タイマXカウント開始 */
}
```

**リスト3-2 LED_PICOPICO 初期設定**

```
/*------------------------------------------------------------------------
 * ID                    : なし
 * 関数名                : led_blink_out
 * 機能                  : p1_1、p1_2に接続されているLEDを点灯⇔消灯させる
 * 引数                  : なし
 * 戻り値                : なし
 * 使用関数    : なし
 *------------------------------------------------------------------------*/
void led_blink_out(void)
{
        p1_2 = ~p1_2;                                   /* p1_2反転 */
        p1_1 = ~p1_1;                                   /* p1_1反転 */
}
```

**リスト3-3 LED_PICOPICO LED出力関数**

## 3.3 サンプルプログラムのワークスペースを開く

HEW を起動します。Windows のスタートメニューより、「プログラム」→「Renesas High-performance Embedded Workshop」の 「High-performance Embedded Workshop」をクリックしてください。

図 3-3 HEW:起動する

「別のプロジェクトワークスペースを開く」にチェックを入れ、「OK」をクリックしてください。

コピーした LED_PICOPICO フォルダ内の「LED_PICOPICO.hws」ファイルを選んで「開く」をクリックしてください。

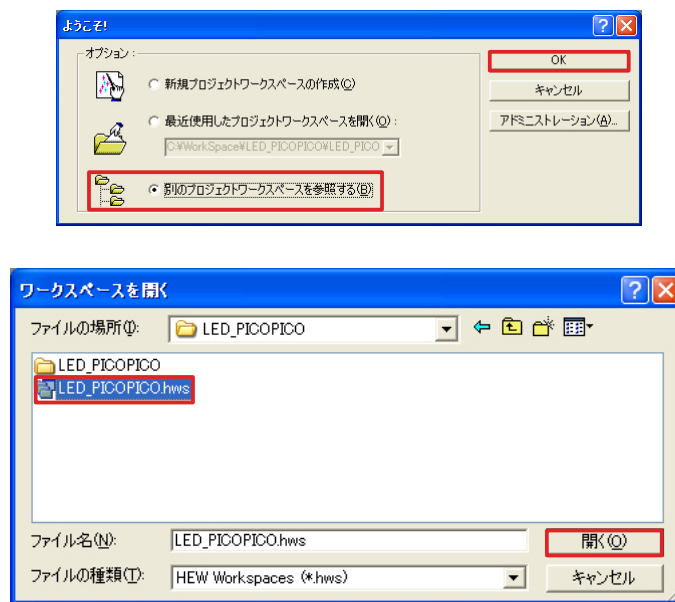



図 3-4 HEW:プロジェクトを開く

LED_PICOPICO ワークスペースは「C: ¥R8C_PROGRAM フォルダ」内に作成したものですので、違う構成のフォルダへコピーした場合は以下のようなメッセージが出ます。「はい」を押してください。このメッセージはフォルダを移動したあと初回のワークスペース参照時のみ出ます。(ここでは C: ¥WorkSpace フォルダへコピーしたこととしています。)

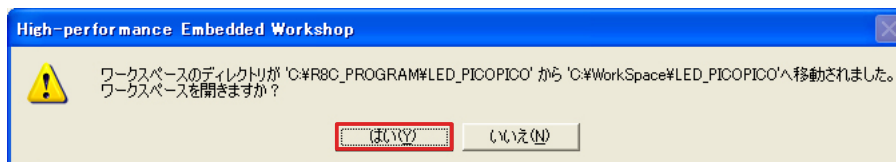



図 3-5 HEW:ワークスペースの移動メッセージ

ワークスペースが開きます。

メニューバー

ビルドコンフィギュレーション、デバッグセッション

ツールバー

ワークスペースウィンドウ
現在開いているワークスペース、プロジェクトの構成を表示します。

エディタウィンドウ
テキストエディタとして使用できます。

アウトプットウィンドウ
HEW による各処理の実行結果が表示されます。

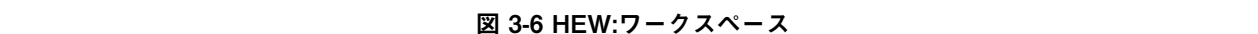
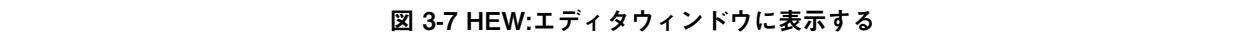
図 3-6 HEW:ワークスペース

「ワークスペースウィンドウ」に表示されているファイルをダブルクリックすると、ファイルの内容が「エディタウィンドウ」に表示されます。

図 3-7 HEW:エディタウィンドウに表示する

※ ビルドコンフィギュレーション、デバッグセッションについては次章「3.4 ビルドコンフィギュレーション、デバッグセッションについて」を参照してください。

# 3.4 ビルドコンフィグレーション、デバッグセッションについて

## ■ ビルドコンフィグレーション

「ビルドコンフィグレーション」には、ビルドを行う際のフェーズ（アセンブラ、コンパイラ、リンカ）の構成や、各フェーズでのオプション設定を保存しています。コンフィグレーションを切り替えることによって、異なるオプションの組み合わせを効率良く切り替えてビルドすることができます。

サンプルプログラムでは各コンフィグレーションを以下のように設定しています。

● Debug
サンプルプログラムでは、評価中の設定としています。このコンフィグレーションでビルドした場合、生成された各ファイルはLED_PICOPICOプロジェクトフォルダ内のDebugフォルダに格納されます。

● Release
サンプルプログラムでは、評価後の最終書き込みの設定としています。LMC30（ロードモジュールコンバータ）の「.ID（IDコードを設定する）」オプションを追加しています。このコンフィグレーションでビルドした場合、生成された各ファイルはLED_PICOPICOプロジェクトフォルダ内のReleaseフォルダに格納されます。

● Debug_R8C_E8_SYSTEM
サンプルプログラムでは、使用していません。このコンフィギュレーションでビルドした場合、生成された各ファイルはLED_PICOPICOプロジェクトフォルダ内のDebug_R8C_E8_SYSTEMフォルダに格納されます。

● Debug_M16C_R8C_FoUSB_UART
サンプルプログラムでは、使用していません。このコンフィギュレーションでビルドした場合、生成された各ファイルはLED_PICOPICOプロジェクトフォルダ内のDebug_M16C_R8C_FoUSB_UARTフォルダに格納されます。

## ■ デバッグセッション

「デバッグセッション」には、デバッグを行う際のデバッギングプラットフォーム（E8エミュレータ等）との接続のための設定等が保存されています。セッションを切り替えることによってデバッギングプラットフォームを簡単に替えることができます。

● DefaultSession
E8エミュレータとは接続せず、ホストPCはスタンドアロンの状態で作業を行います。

● SessionR8C_E8_SYSTEM
E8エミュレータとの接続のための設定が登録されています。このセッションに切り替えるとE8エミュレータとの接続ウィザードが開始され、ホストPCはE8エミュレータを介してターゲットシステムと接続されます。（セッション切り替え時に、自動的に接続ウィザードが開始させないように設定することもできます。）

● SessionM16C_R8C_FoUSB_UART
シリアルインターフェイスで接続するための設定が登録されています。

## 3.5 E8 エミュレータと接続する

ホスト PC に E8 エミュレータを接続し、E8 エミュレータをターゲットボードに接続してください。

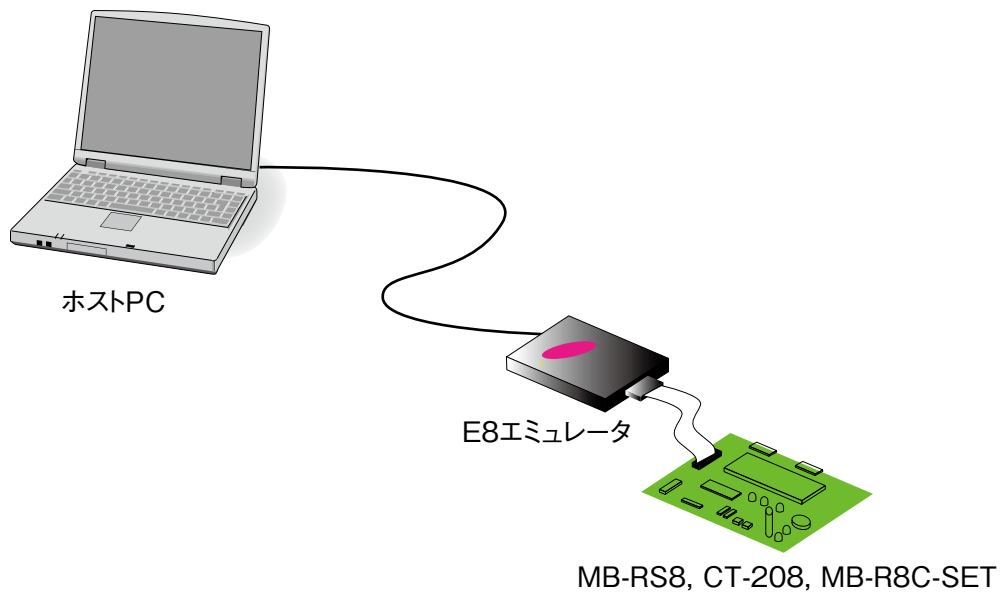



**図3-8 ホストPCとE8エミュレータとターゲットシステムとの接続**

### E8エミュレータとの接続時の注意事項（重要）

ホスト PC、E8 エミュレータ、ターゲットシステムをそれぞれ接続して評価中は、HEW とターゲットチップのファームウェアが頻繁に通信を行っており、HEW のコマンドによってはフラッシュメモリの消去 / 書き込みを行っている場合があります。各接続ケーブルの取り外し、ターゲットマイコンのリセットについては以下についてご注意ください。

- ホスト PC と E8 エミュレータ、E8 エミュレータとターゲットシステムを接続しているケーブルをはずす場合は、かならず HEW より「接続解除」コマンドを実行するか、HEW を終了してください。さらにホスト PC から E8 エミュレータの USB ケーブルをはずす場合は、Windows で「ハードウェアの安全な取り外し」を行い、E8 エミュレータのパワーインジケータが消灯しているのを確認してからはずしてください。
- ユーザープログラムの評価中は、ホスト PC、ターゲットシステムのパワーオフ、リセットはしないでください。ターゲットマイコンの状態を初期化したい場合は、HEW の「デバッグ」メニュー→「CPU のリセット」コマンドで行ってください。

ターゲットシステム上の各スイッチ、設定ジャンパを以下のように設定してください。設定のあと、かならずターゲットマイコンをリセットしてください。

■ MB-RS8 の場合

・モード切り替えスイッチ：RUN

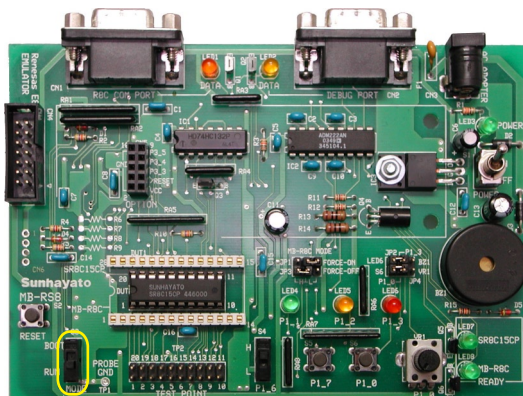



**図3-9 MB-RS8のE8エミュレータ接続のための設定**

■ CT-208 の場合

・モード切り替えスイッチ：RUN

・設定ジャンパ：JP1 → E8

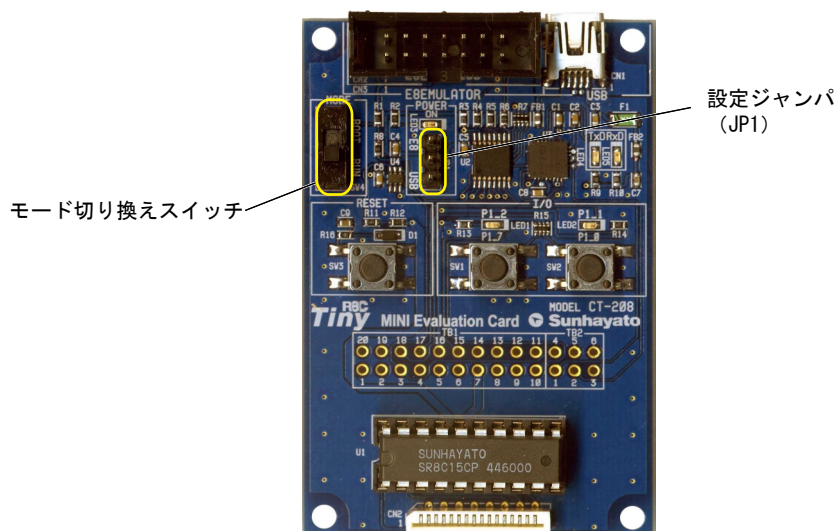



**図3-10 CT-208のE8エミュレータ接続のための設定**

■ MB-RS8-SET 他、オリジナルターゲットシステムの場合

ターゲットマイコンがユーザーモードで起動するように設定してください。

デバッグセッションを「DefaultSession」から「Session_R8C_E8_System」へ切り替えます。

**図3-11 HEW:「SessionR8C_E8_SYSTEM」に切り替える**

E8 エミュレータとの接続ウィザードが始まり、「Select Emulator Mode」ウィンドウが開きます。E8 エミュレータは「Select Emulator Mode」ウィンドウで以下の 3 つの動作モードを選択できます。

■ Download emulator firmware
ターゲットマイコンにファームウェアを書き込んだあと起動します。ターゲットマイコンにファームウェアが存在しない場合に選択します。

■ Does not download emulator firmware
ターゲットマイコンにファームウェアを書き込まずに起動します。ターゲットマイコンに既にファームウェアが存在する場合に選択します。

■ Writing Flash memory
ターゲットマイコンにファームウェアを書き込まずに起動します。E8 エミュレータをフラッシュメモリライタとして使用する場合に選択します。

ここでは初めて E8 エミュレータとターゲットシステムを接続するものとして、「Download emulator firmware」を選択します。以下のように設定してください。

- Device: R5F21154
- Mode:　Download emulator firmware にチェック

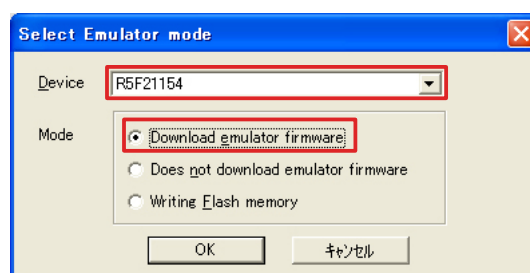



**図3-12 HEW: Select Emulator Mode**

サンハヤト R8C/Tiny マイコンシリーズの書き込みボードは 5V を供給する仕様になっていますので、ターゲットシステムに E8 エミュレータから電源を供給する場合は､｢Power supply is carried out (MAX 300mA)」にチェックを入れて、｢5.0V」をチェックしてください。｢OK」を押すと、電源が供給されます。

ターゲットシステムに E8 エミュレータから電源を供給しない場合は、｢Power supply is carried out (MAX 300mA)」にチェックを入れないで「OK」をクリックし、その後で電源を投入してください。

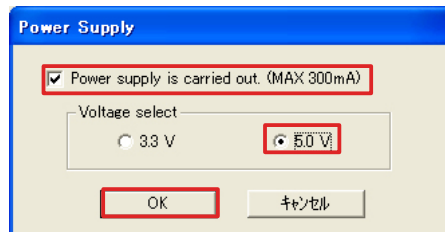



**図3-13 HEW: Power Supply**

ファームウェアを書き込む領域を指定します。SR8C15CP マイコン（あるいは R5F21154）の場合、｢Data Flash Area」をチェックすると 2400h ～ 2BFFh に、User Flash Area をチェックすると C000h ～ C7FFh に書き込まれます。

サンプルプログラムではプログラムの開始アドレスを C000h に設定しているので､｢Data Flash Area」をチェックしてください。次回接続する際に｢Does not download emulator firmware｣モードを選択した場合､ファームウェアが書き込まれている領域を指定する必要がありますので、｢Data Flash Area」か「User Flash Area」のどちらに書き込んだかを覚えておいてください。

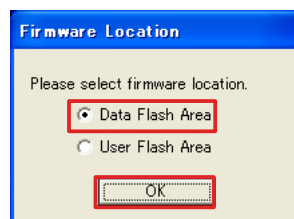



**図3-14 HEW: Firmware Location**

## 2回目以降のE8エミュレータとの接続について

一度ターゲットマイコンにファームウェアを書き込んだら、次回の接続ウィザードからは「Does not download emulator firmware」モードで起動してください。「Writing Flash memory」で機械語ファイルを書き込むか、他のフラッシュメモリ書き込みツールで書き込む等してファームウェアを消去しないかぎり、一度書き込まれたファームウェアは有効です。

指定した領域にファームウェアが書き込まれます。

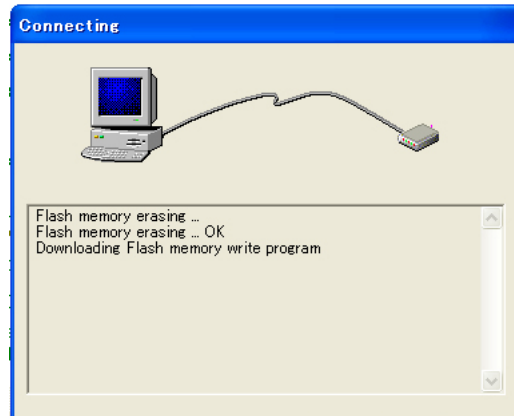



**図3-15 HEW: ファームウェアの書き込み**

ファームウェアが書き込まれると、E8 エミュレータを介してホスト PC の HEW とターゲットマイコンに書き込まれたファームウェアとの通信が開始します。ツールバーのデバッグコマンドのアイコンがアクティブになり、エディタウィンドウの左側に実アドレス、ブレークポイント、PC（プログラムカウンタ）の位置を表示するスペースが追加されます。（アドレスはプログラムが書き込まれた後に表示されます。）

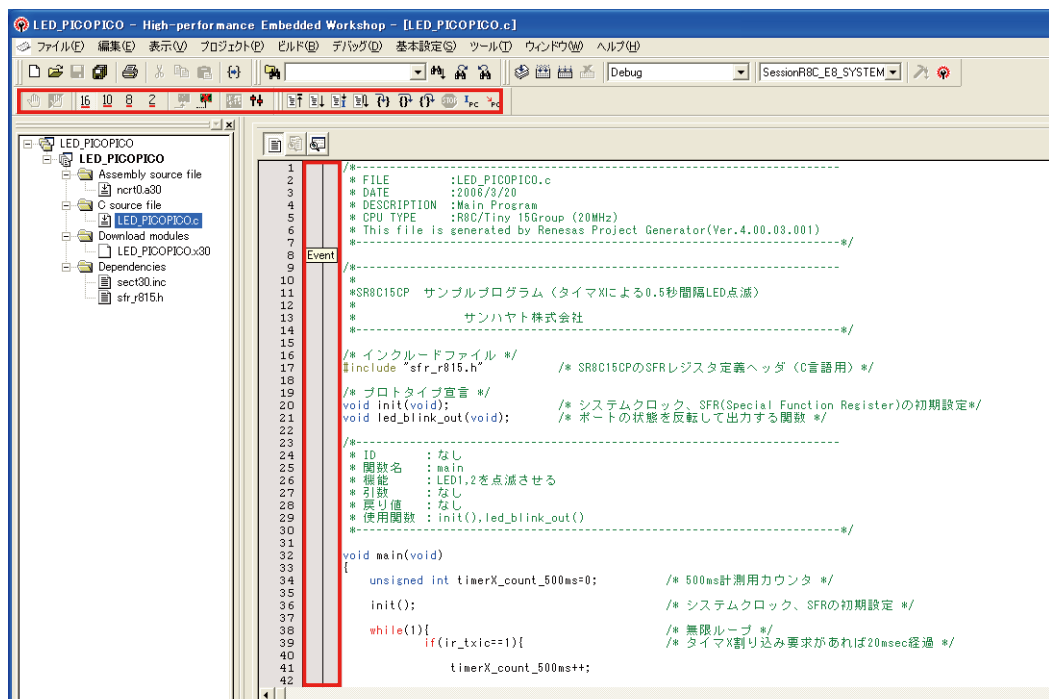



**図3-16 HEW: E8エミュレータと接続後のワークスペース**

## 3.6 プログラムをビルドする

メニューバーの「ビルド」メニューの「すべてをビルド」をクリックすると、ビルドが実行され、各フェーズでのメッセージがアウトプットウィンドウに表示されます。(「すべてをビルド」アイコンをクリックしてもビルドを実行できます。)

図3-17 HEW:ビルドする

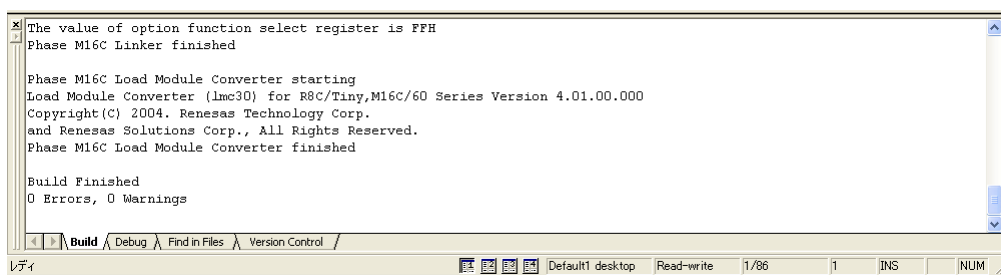



図3-18「アウトプットウィンドウ」に表示されたビルド結果

## 3.7 プログラムをダウンロードする

「デバッグ」コマンドの「ダウンロード」より C: ¥Hew ¥Program ¥LED_PICOPICO ¥led_picopico…を選んでください。（コピーしたフォルダ名先のファイル（.x30）が表示されます。）（図 3.16）あるいは、ワークスペースウィンドウに表示されている LED_PICOPICO.x30 を選択して右クリックし、ポップアップメニューより「ダウンロード」をクリックしてもダウンロードできます。（図 3.20）

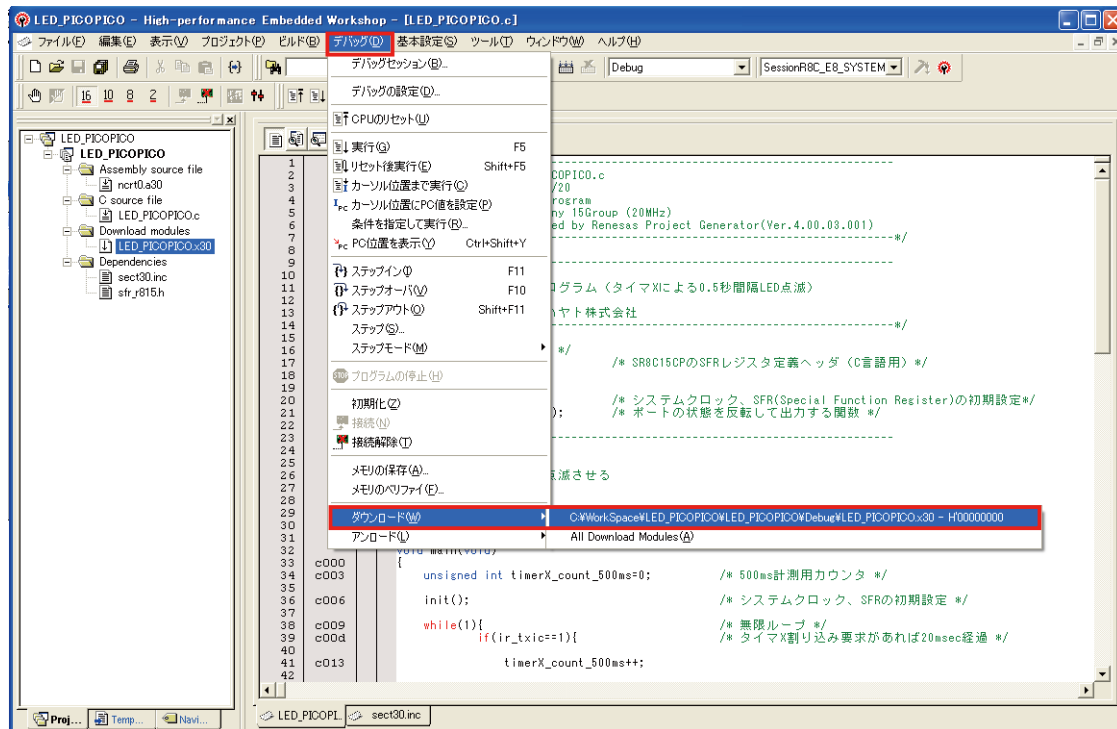



**図3-19 HEW:アブソリュートモジュールファイルのダウンロード1**

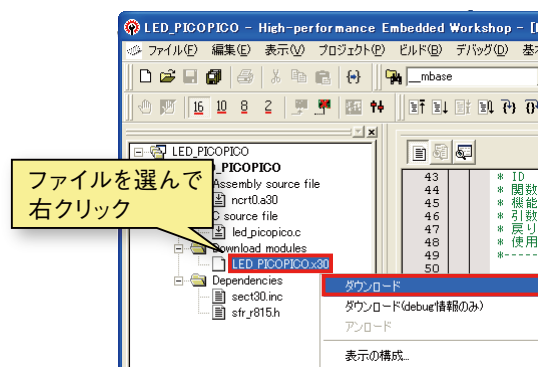



**図3-20 HEW:アブソリュートモジュールファイルのダウンロード2**

| プログラムのダウンロード中にホスト PC、ターゲットシステムのパワーオフ、リセットはしないでください。 |
|---|

アブソリュートモジュールファイルが書き込まれると、エディタウィンドウの左側に実アドレスが表示されます。

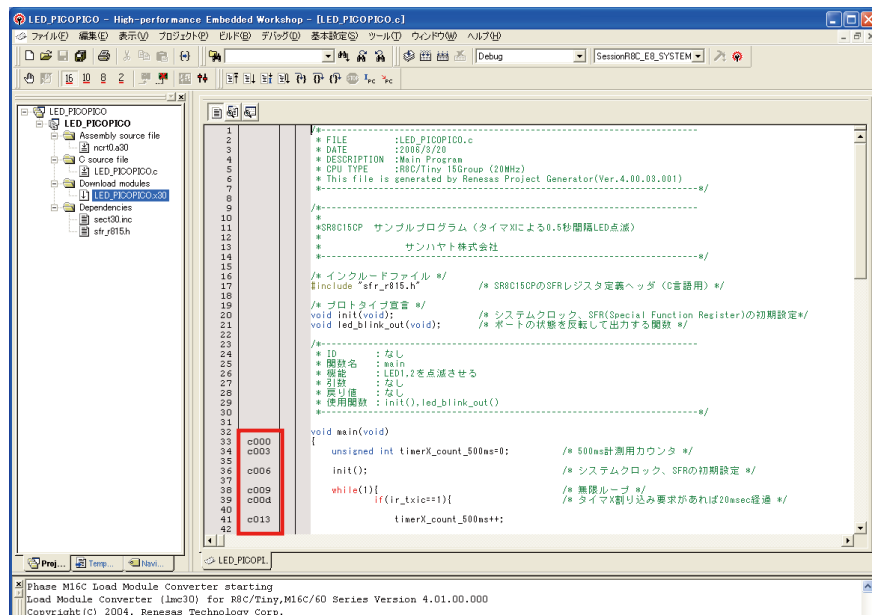



図3-21 HEW: エディタウィンドウに表示された実アドレス

エディタウィンドウ左上の表示切り替えボタンを押すと、「ソースモード（C 言語のみ）」から「混合モード（C 言語とアセンブリ言語）」、「逆アセンブリモード（アセンブリ言語のみ）」に切り替えることができます。（E8 エミュレータとターゲットシステムが接続されていて、ターゲットマイコンにユーザープログラムが書き込まれているときのみ切り替え可能です。）

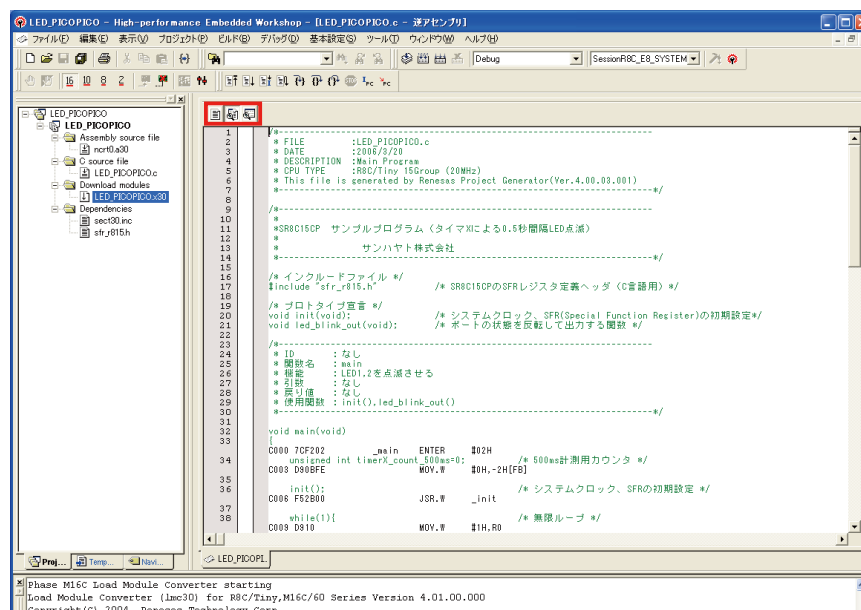



図3-22 HEW: エディタウィンドウ表示（混合モード）

## 3.8 ターゲットマイコンをリセットする

「デバッグ」メニューの「CPU のリセット」か、ツールバーの「CPU のリセット」アイコンをクリックすると、ターゲットマイコンがリセットされます。このコマンドにより内蔵 I/O レジスタの初期化、PC（プログラムカウンタ）の初期化（リセットベクタに設定されているアドレスが設定されます）が行われます。プログラム書き込み後は、必ずターゲットマイコンをリセットしてからプログラムを実行してください。

また、ターゲットマイコンのリセットは必ず HEW のリセットコマンドで行い、ターゲットシステムから直接リセットしないでください。

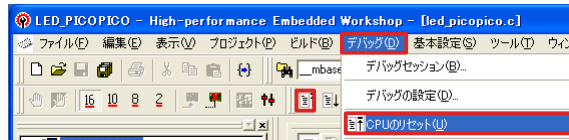



図3-23 HEW: CPUのリセット

## 3.9 プログラムを実行する

「デバッグ」メニューの「実行」か、ツールバーの「実行」アイコンをクリックすると、プログラムが実行されます。「リセット後実行」をクリックすると、リセット後にプログラムが実行されます。基板上の LED（P1_1,P1_3）が交互に点滅するのを確認してください。

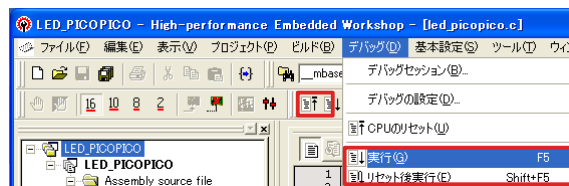



図3-24 HEW: プログラムの実行

## 3.10 プログラムを停止する

「デバッグ」メニューの「プログラムの停止」か、ツールバーの「STOP」アイコンをクリックすると、プログラムは停止し、現在の PC（プログラムカウンタ）の位置が黄色の矢印で示されます。

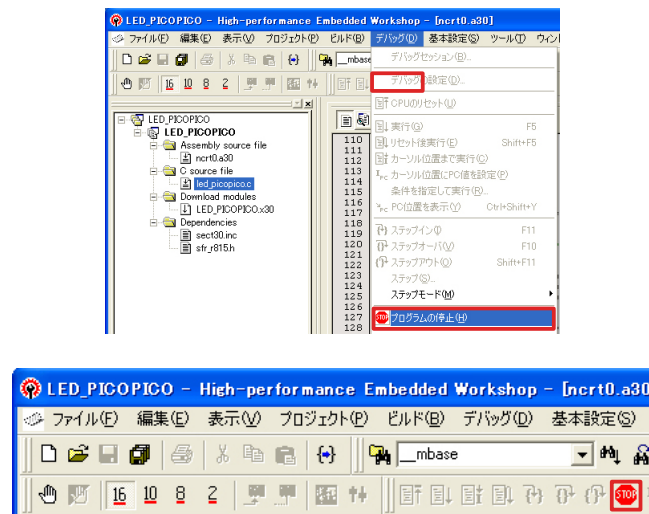



**図3-25 HEW: プログラムの停止**

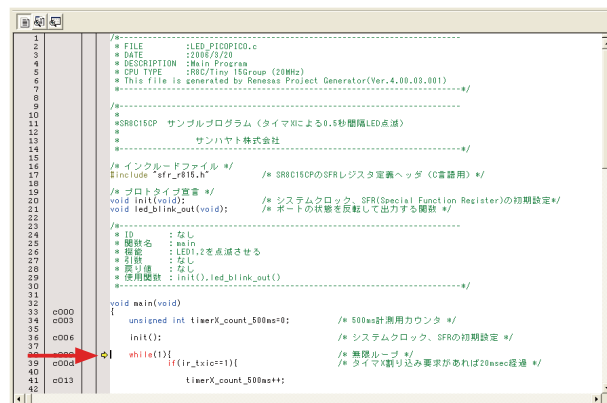



**図3-26 HEW: プログラム停止後のPC**

## 3.11 レジスタ、I/O を表示する

「表示」メニューの「CPU」で、「レジスタ」あるいは「I/O」をクリックしてください。

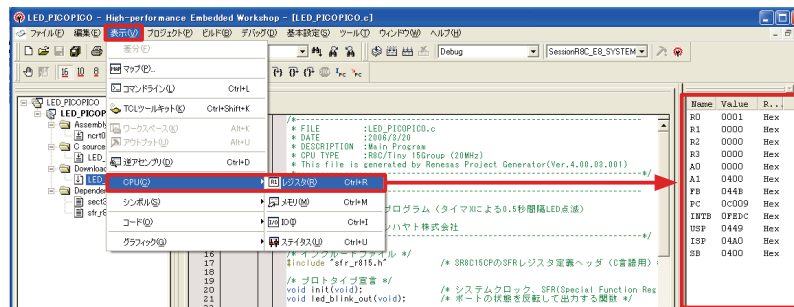



図3-27 HEW:レジスタの表示

各ウィンドウエリアで右クリックし、ポップアップウィンドウの「ドッキングビュー」のチェックを外すと、ウィンドウをフローティングさせることができます。

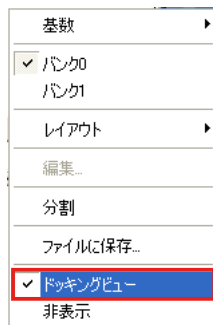



図3-28 ウィンドウのフローティング

## 3.12 PC の値を変更する

ソースライン上のカーソルの位置のアドレスを、PC（プログラムカウンタ）に設定することができます。プログラム実行を開始したい行にカーソルを置き､｢デバッグ｣メニューの｢カーソル位置の PC 値を設定｣か､ツールバーの｢カーソル位置に PC 設定｣アイコンをクリックしてください。PC に設定された行に黄色い矢印が表示されます。

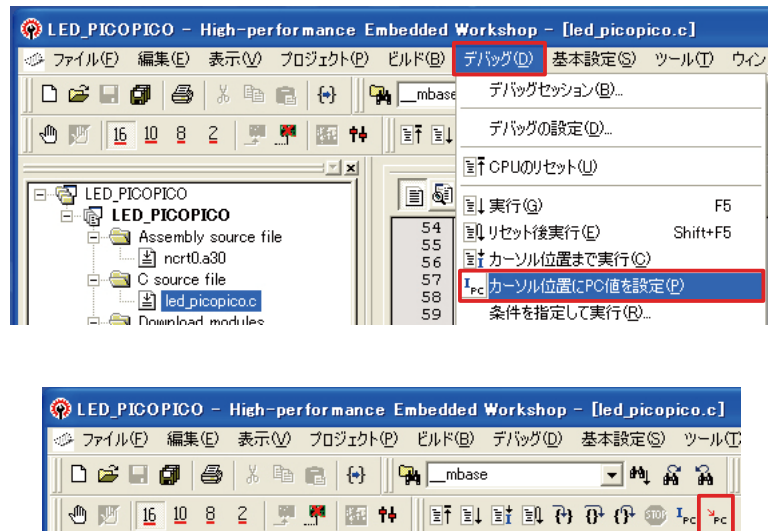



**図3-29 HEW: カーソル位置にPC値を設定する**

## 3.13 S/W ブレークポイントを設定（解除）する

プログラムを停止させたい行にカーソルを置いた状態で、“F9”キーを押すか、リスト左のグレーの部分をダブルクリックしてください。ブレークポイントが設定された行に赤いマークが付きます。プログラムを実行させると、ブレークポイントが設定された行（命令）のひとつ前の命令までを実行して止まります。

S/W ブレークポイントは最大 255 ポイントまで設定できます。S/W ブレークポイントが設定された箇所にカーソルを置いた状態で“F9”キーを押すか、ブレークポイントを直接ダブルクリックすると S/W ブレークポイントが解除されます。

※ S/W ブレークポイントは、設定されたアドレスの命令を BRK 命令に書き換えることで実現しています。（フラッシュメモリの書き換えが行われます。）

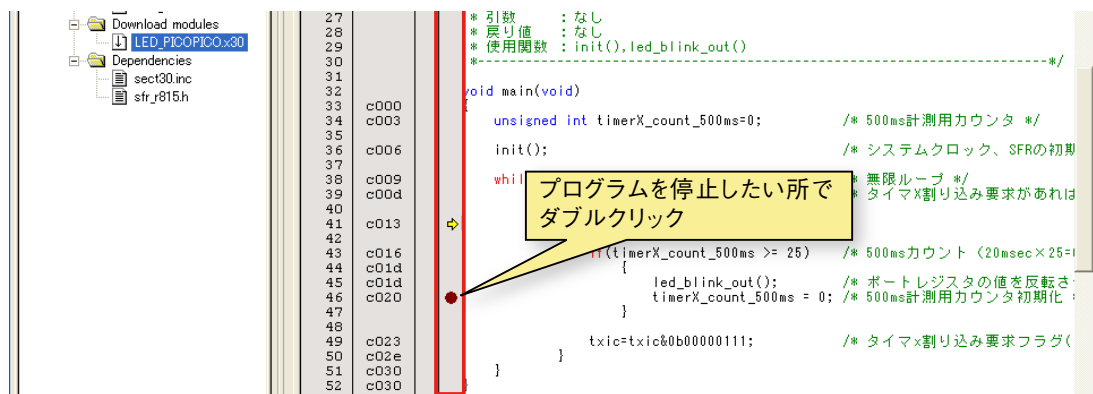



図3-30 HEW: S/Wブレークポイントの設定

## 3.14 アドレス一致ブレークポイントを設定（解除）する

プログラムを停止させたい行のリスト左のグレーの部分を、ダブルクリックをしてください。アドレス一致ブレークポイントが設定された行に青いマークが付きます。プログラムを実行させると、ブレークポイントが設定された行（命令）のひとつ前の命令までを実行して止まります。設定したアドレスブレークポイントを直接ダブルクリックすると解除されます。

※ アドレス一致ブレークは、ターゲットマイコンのアドレス一致割り込み機能を使用しています。SR8C15CP マイコンの場合はアドレス一致ブレークを最大 4 ポイントまで設定できます。

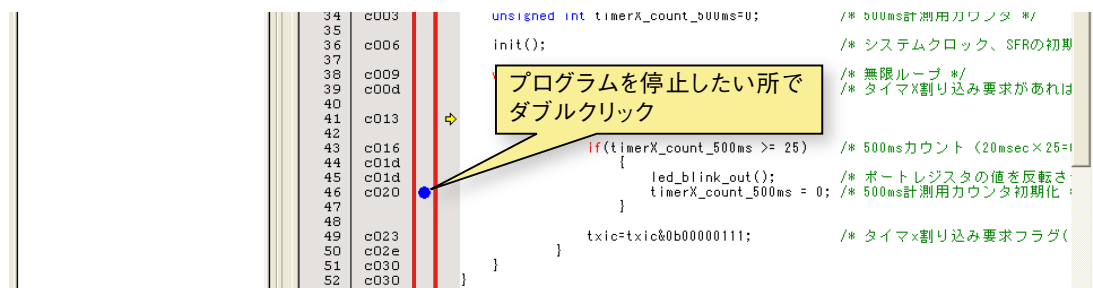



図3-31 HEW: アドレス一致ブレークポイントの設定

## 3.15 シングルステップを実行する

シングルステップは1行(あるいはアセンブリ言語での1命令)を実行します。C言語での1行を実行するか(ソースモード)、アセンブリ言語での1命令を実行するか（アセンブリソースモード）は、「デバッグ」メニューの「ステップモード」で設定します。「自動選択」(デフォルト)に設定すると、現在アクティブになっているエディタウィンドウがソースウィンドウの場合はソースレベルの1行、アセンブリウィンドウであれば、アセンブリ言語の1命令を実行します。

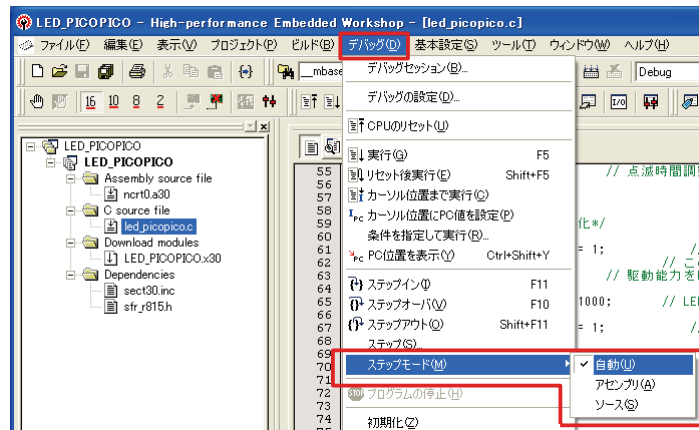



**図3-32 HEW: ステップモードの設定**

ステップを実行する行（命令）が関数を呼び出ししている場合の行（命令）の実行の方法として、以下の3つから選択できます。

- **ステップイン**
  関数を呼び出す命令のみを実行します。次にステップイン実行した場合、関数の中の次の命令を実行します。

- **ステップオーバ**
  関数を呼び出す命令と関数中の命令を全て（および、さらにその中で呼び出していて実行する可能性のある関数の中の命令全て）実行してから、呼び出し元の、関数を呼び出す命令の次の命令を実行する前で停止します。

- **ステップアウト**
  関数内の確認したい命令の実行が終了した場合や、誤って関数にステップインした場合、ステップアウトを実行することで、関数内の残りの命令を実行せずに呼び出し元の関数に戻る事が出来ます。

「デバッグメニュー」から「ステップイン」、「ステップオーバ」、「ステップアウト」をそれぞれクリックするか、ツールバーの各アイコンをクリックしてください。

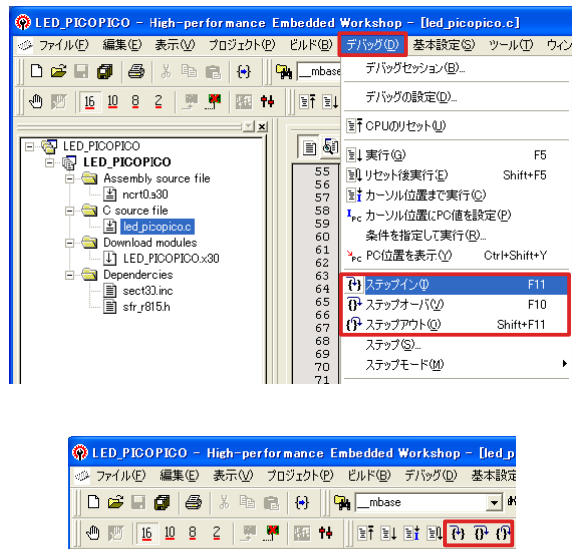



図3-33 HEW: ステップ実行

## 3.16 複数のステップを実行する

「プログラムステップ」ウィンドウに設定することで､一度に複数のステップ実行ができます。ステップの回数、実行する命令（行）の間の時間設定（1 ～ 3 秒まで、0.5 秒間隔で設定できます）と、ステップオーバ、ソースレベルステップをそれぞれ選択できます。

「デバッグ」メニューの「ステップ」を選択し､「プログラムステップ」ウィンドウで各設定をしてください。「OK」を押すと設定した複数ステップが実行されます。

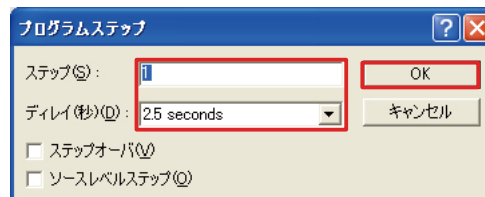
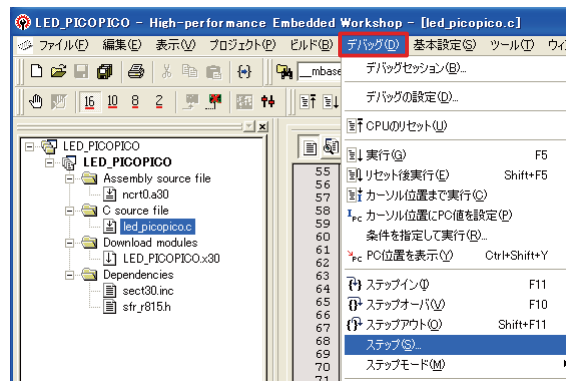



**図3-34 HEW: 複数のステップ実行**

## 3.17 接続を解除する

「デバッグ」メニューの「接続解除」か､ツールバーの「接続解除」アイコンをクリックしてください。E8 エミュレータをターゲットシステム、ホスト PC から外す場合はかならず「接続解除」をしてから行ってください。

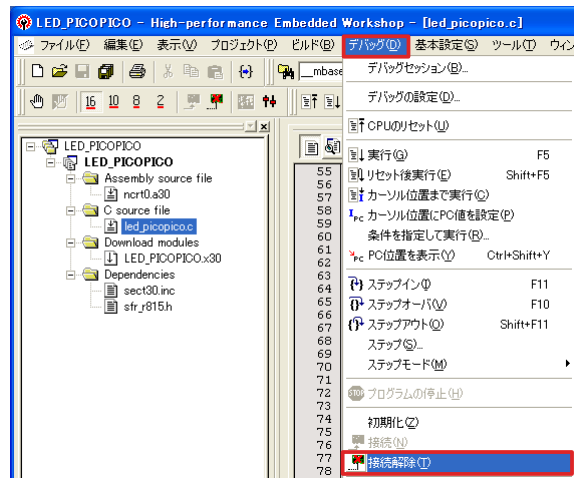



図 3-35 HEW: 接続解除

## 3.18 接続を再開する

接続を解除したあとに、再び接続するには､「ビルド」メニューの「デバッグ」→「接続」か、ツールバーの「接続」アイコンをクリックしてください。接続ウィザードが開始されます。

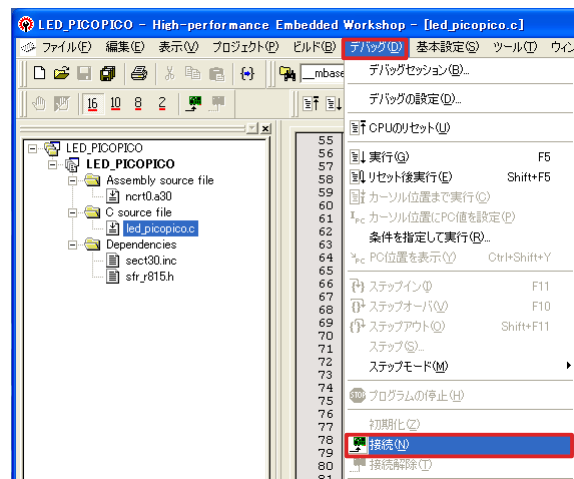



図3-36 HEW: 接続を再開する

## 3.19 フラッシュ書き込みモードで書き込んでみる

開発の最終リリースを想定して、ファームウェアを消去し、ユーザープログラムのみをフラッシュメモリに書き込んでみましょう。

### ■ Releaseコンフィギュレーションでビルドする

デバッグセッションが「SessionR8C_E8_SYSTEM」であれば「DefaultSession」に切り替え、ビルドコンフィギュレーションを「Release」に切り替えて「すべてをビルド」を実行してください。

サンプルプログラム「LED_PICOPICO」では「Release」コンフィギュレーションで LMC30（ロードモジュールコンバータ）の .ID オプションを設定しています。（ID コードにすべて FFh を設定します。）

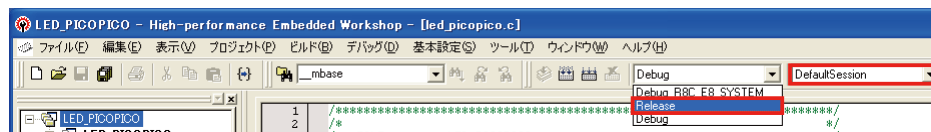



図3-37 HEW: コンフィギュレーションの切り替え

**IDコードについて… 概要**

ID コードとは、フラッシュメモリの内容のセキュリティのための 7 バイトのコードをいいます。あらかじめフラッシュメモリの決められた番地（R8C/Tiny シリーズの場合は FFDFh, FFE3h, FFEBh, FFEFh, FFF3h, FFF7h, FFFBh）にコードを書き込んでおき、次のフラッシュメモリの書き換え時に ID コードが照合しなければ、フラッシュメモリにアクセスできない仕組みになっています。

マイコン出荷時はブランク状態で ID コードとしてどのような値を入力しても書き込み可能になっています。プログラムや ID オプションで特に ID コードを設定せずにプログラムを書き込んだ場合はすべて FFh が設定されます。

ID コードは前回設定した値を忘れてしまうとフラッシュメモリの書き換えができなくなるので、評価中はすべて FFh など覚えやすい値を設定するのがよいでしょう。

## ■ モトローラSフォーマットファイルをダウンロードモジュールに追加する

ユーザープログラムのみを書き込む場合はモトローラＳフォーマットファイル（.mot）を書き込みます。ビルドによってモトローラＳフォーマットファイル（LED_PICOPICO.mot）はすでに生成されていますが、HEW でモトローラＳフォーマットファイルを書き込むために、ダウンロードモジュールとしてモトローラＳフォーマットファイルを追加登録しておく必要があります。

「デバッグ」メニューの「デバッグの設定」をクリックし、右側設定タブを「ターゲット」にして以下のとおり設定してください。

- 設定対象となるセッション：「SessionR8C_E8_SYSTEM」
- ターゲット：「R8C_E8_SYSTEM」
- デバッグ対象フォーマット：「S-Record」

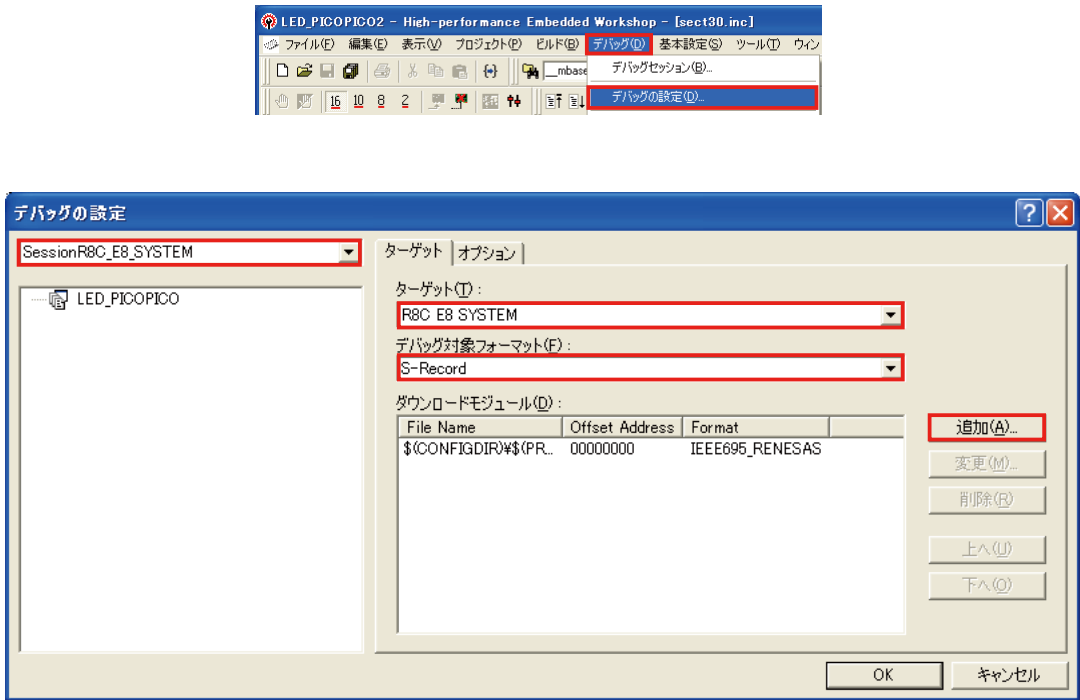
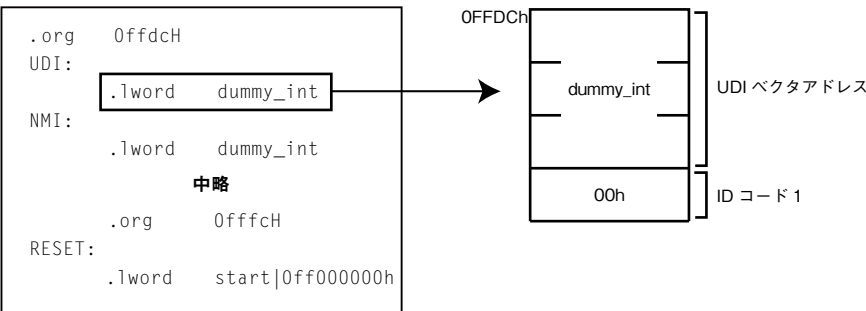



図3-38 HEW: デバッグの設定

「ダウンロードモジュール」の「追加」をクリックすると、ダウンロードモジュールの設定画面が開きます。

### IDコードについて… 設定方法

IDコードの設定方法には、①プログラムリストで設定する、②アセンブラ擬似命令 .ID で設定する、③ LMC30 の ID オプションで設定するの３つがあります。全ての方法で設定した場合、①＜②＜③の順で設定内容が優先されます。

プログラムリストで設定する場合は、IDコード設定番地が固定ベクタテーブルの間にあるので、セクション定義ファイルで設定することができます。たとえばセクション定義ファイル section30.inc での設定では、アセンブラ擬似命令 .lword で４バイト設定される内の上位１バイトには 00h が設定されるので、結果的に ID コードとして 00h が設定されることになります。

以下のように設定してください。

- オフセット：0000000000
- フォーマット：S-Record

ファイル名は以下の順で設定してください。

①「▶」をクリックして「コンフィグレーションディレクトリ」をクリックする。
②手入力で「¥」を入れる。
③「▶」をクリックして「プロジェクト名」をクリックする。
④手入力で「.mot」を入れる。
⑤"$(CONFIGDIR)¥$(PROJECTNAME).mot" と設定されていることを確認して「OK」をクリックする。

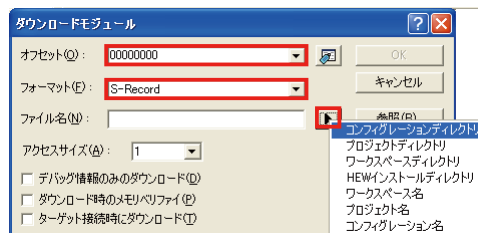



図3-39 HEW: ダウンロードモジュールの指定

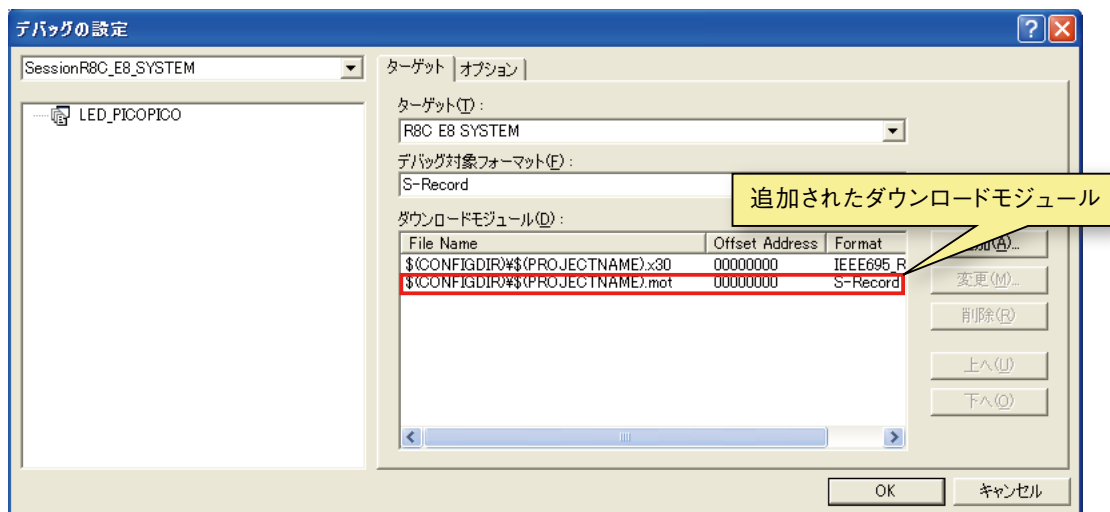



図3-40 HEW: 追加されたダウンロードモジュール

## ■ モトローラSフォーマットファイルを書き込む

デバッグセッションを「DefaultSession」から「SessionR8C_E8_SYSTEM」に切り替えます。

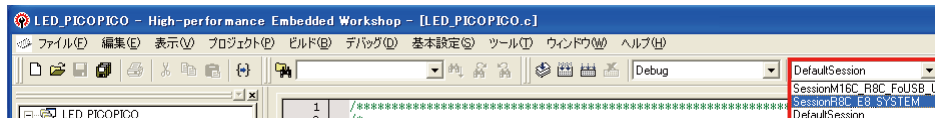



**図3-41 HEW: コンフィギュレーションを「Release」に切り替える**

接続ウィザードが開始します。E8 エミュレータをフラッシュメモリライタとして使用するので、「Select Emulator Mode」では「Writing Flash memory」にチェックを入れてください。

**図3-42 HEW: エミュレータ動作モードの設定**

電源の設定をしてください。

**図3-43 HEW: 電源の設定**

「ID Code verification」(ID コードの認証) では、"FF FF FF FF FF FF FF" と設定してください。(実際に入力する時は、数字の間にスペースを入れないでください。)

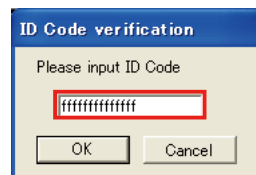



**図3-44 HEW: IDコードの認証**

### IDコードについて… E8エミュレータ起動後の設定値

E8 エミュレータで「Download emulator firmware」、「Does not download emulator firmware」モードで起動後はファームウェアにより ID コードはすべて FFh が設定されます。

E8 エミュレータで「Writing Flash memory」モードで起動し、ユーザープログラムを書き込んだ場合は、ユーザープログラムで設定した ID コードが設定されます。(「Writing Flash memory」モードでサンプルプログラム「LED_PICOPICO」、「LED_PICOPICO2」を書き込んだ場合はすべて FFh が設定されます。)

プロジェクトスペースの「LED_PICOPICO.mot」を右クリックして「ダウンロード」をクリックしてください。フラッシュメモリが一度消去され、「LED_PICOPICO.mot」が書き込まれます。

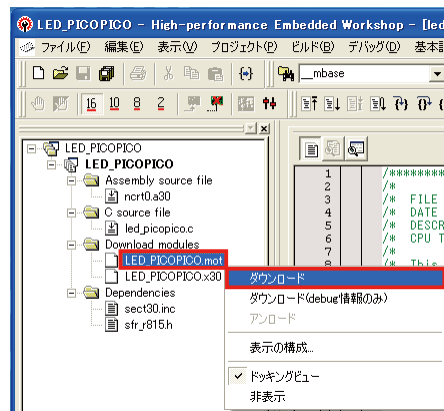



**図3-45 HEW: LED_PICOPICO.motのダウンロード**

書き込み終了のメッセージが出ますのでそれ「OK」をクリックしてください。

**図3-46 HEW: プログラム書き込み終了メッセージ**

※ E8 エミュレータを「Writing Flash memory」で接続した場合、HEW のプログラム、レジスタ、メモリウィンドウ等の内容はすべて実際の数値ではなく、ダミーデータが表示されています。各データの変更は行わないでください。

プログラムのダウンロード中にホスト PC、ターゲットシステムのパワーオフ、リセットはしないでください。

## ■ ターゲットボードで確認する

「3.21 E8 エミュレータをはずす」を参考に、ターゲットシステムから E8 エミュレータを取りはずし、ターゲットシステムに電源を投入して、書き込まれたプログラムが動作するか確認してください。

CT-208 の場合は設定ジャンパ JP1 を USB 側に設定した後に、USB ケーブルでホスト PC から電源を供給してください。

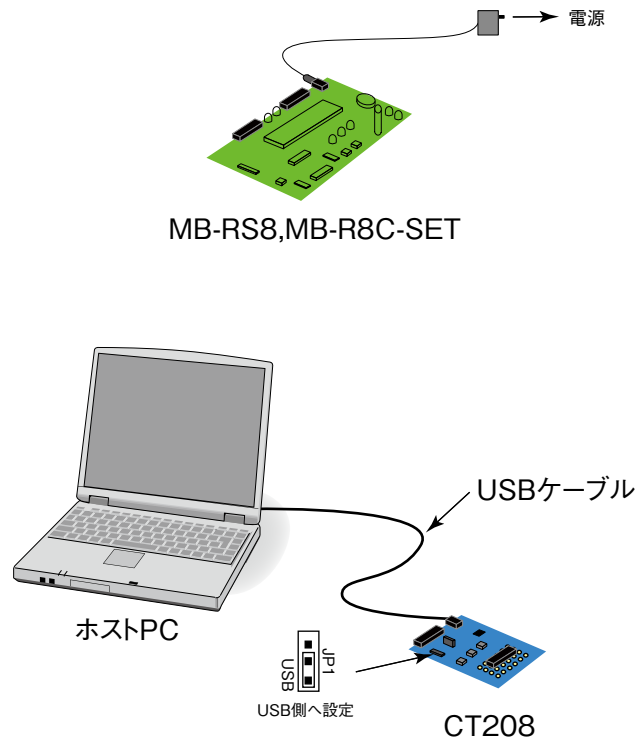



**図3-47 HEW: プログラムの動作確認**

オリジナルターゲットシステムの場合はターゲットマイコンに電源を投入してください。

## 3.20 ワークスペースを閉じる

「ファイル」メニューの「ワークスペースを閉じる」をクリックするか､HEW 画面右上のグレーの「×」印をクリックするとワークスペースが閉じます。閉じる際に「すべてのエディタウィンドウを閉じる」か､「現在のワークスペースの状態を保存する」かを尋ねますので、それぞれ「OK」､「保存する」をクリックしてください。

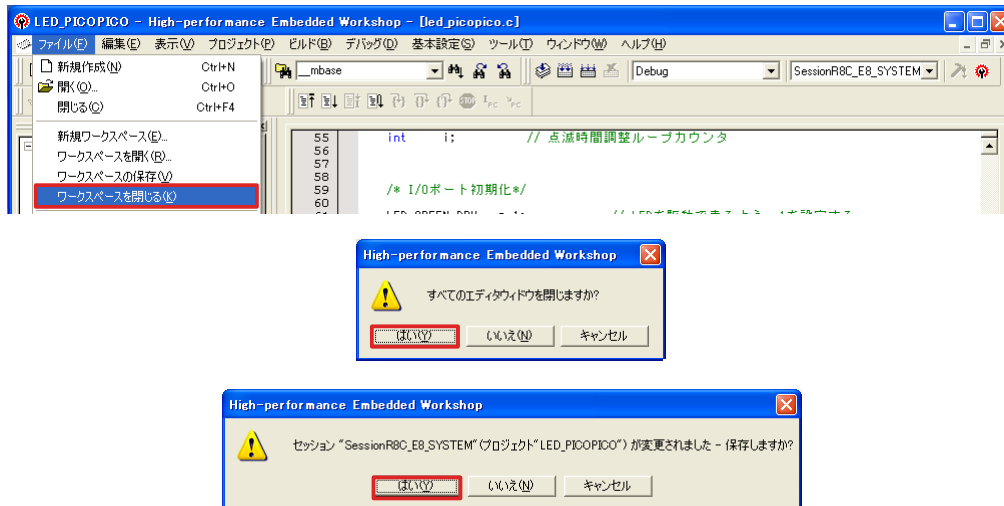



**図3-48 HEW: ワークスペースを閉じる**

ワークスペースを一度閉じると、次回開けるときは前回閉じた時点でのセッションから始まります。

サンプルプログラムの場合、「DefaultSession」の状態でワークスペースを閉じると、次にワークスペースを開くと「DefaultSession」で始まります。「SessionR8C_E8_SYSTEM」のセッションで閉じると、次回は「SessionR8C_E8_SYSTEM」で始まり、すぐに接続ウィザードが開始されます。

## 3.21 E8 エミュレータをはずす

以下の手順にしたがってターゲットシステムから E8 エミュレータをはずしてください。

① HEW の「デバッグ」メニューの「接続解除」コマンドを実行するか、HEW を終了する。「接続解除」コマンドを実行した場合は HEW のアウトプットウィンドウに「Disconnected」とメッセージが出るのを確認する。

② Windows タストレイの「ハードウェアの安全な取り外し」をクリックし､「E7/E10 Adapter」を選択して「停止」をクリックする。さらに次のウィンドウで「E7/E10 Adapter」を選択して「OK」をクリックする。E8 エミュレータのインジケータが消灯しているのを確認する。

③ ターゲットシステム⇔E8 エミュレータのケーブルと、E8 エミュレータ⇔ホスト PC の USB ケーブルを取りはずす。

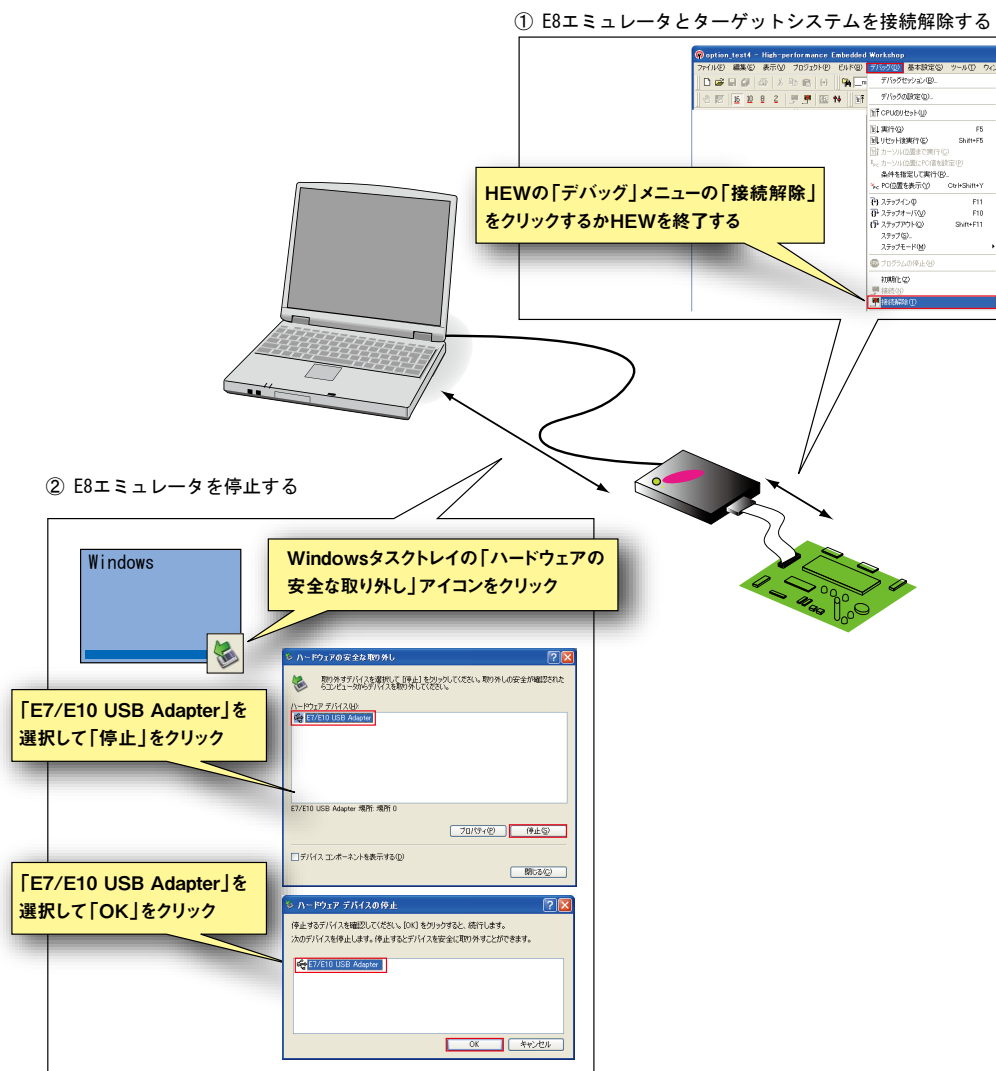



**図3-49 E8エミュレータの取りはずし**

# 4. プログラムをつくろう

## 4.1 ワークスペース、プロジェクト作成ウィザード

HEW でプログラムを作成する場合、まずワークスペース、プロジェクトを作成します。ワークスペース、プロジェクトの作成は、ウィザードにしたがって行います。ウィザードでは、ワークスペース、プロジェクトの名前以外に、プロジェクトに含むスタートアップファイル、ソースファイルの生成、登録や、ターゲットマイコンの品種に応じてコンパイラ等のオプションの設定、使用するデバッグ環境のための設定を付加します。

ここでは、サンプルプログラム「LED_PICOPICO」の内容を改変した新しいプログラム「LED_PICOPICO2」を作成する例をあげて説明します。

## 4.2 新規ワークスペース、プロジェクトの作成

ワークスペース、プロジェクトを作成します。「ファイル」メニューの「新規ワークスペース」をクリックしてください。

図 4-1 HEW: 新規ワークスペース

新規プロジェクトワークスペースウィンドウが開きます。

ここでは、C:¥WorkSpace 以下に LED_PICOPICO2 というワークスペースを作成することとします。(HEW では、ワークスペース、プロジェクト名に英数文字とアンダーバー "_" が使用できます。)

- ワークスペース名、プロジェクト名：LED_PICOPICO2
- CPU 種別：「M16C」

  CPU 種別を「M16C」に選択するとツールチェインは「Renesas M16C Standard」、プロジェクトの種類は「Application」となります。

設定が終わったら「OK」をクリックしてください。

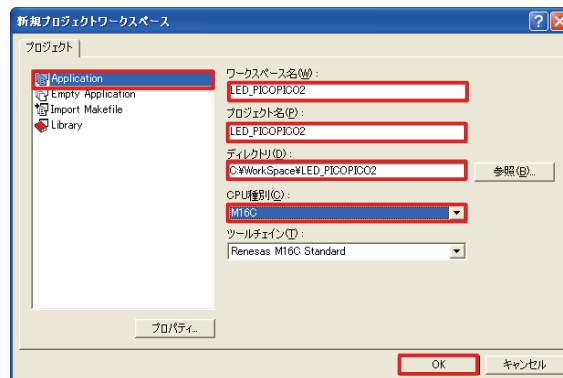



**図4-2 HEW: 新規プロジェクトワークスペース**

「Select CPU Toolchain version」ウィンドウではターゲットマイコンのシリーズを設定します。この設定は以降の各設定やコンパイラなどのオプションに影響しますので間違えないようにしてください。

「CPU Sereis:」を「R8C/Tiny」に、「CPU Group:」を「15 (ROM16K)」にしてください。「Next」をクリックしてください。

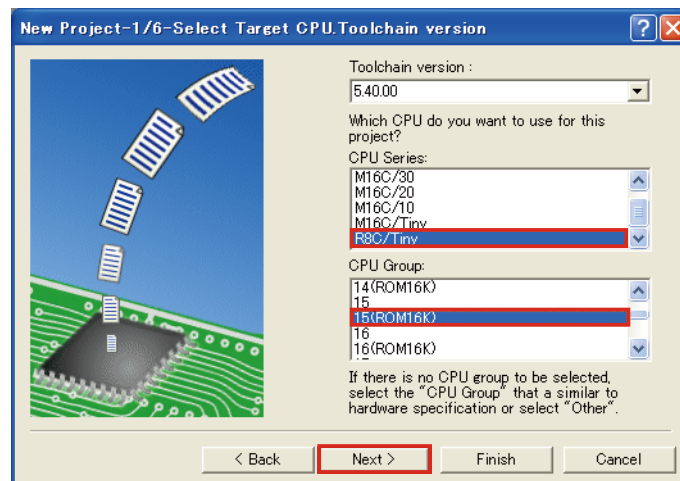



**図4-3 HEW: Select CPU Toolchain version**

## HEWでのファイルの扱いについて

HEW でのソースファイルの作成の仕方は、①プロジェクト新規作成時に他のプロジェクトからか、標準のファイルをコピーする、②プロジェクト作成後にファイルを新規作成する、③プロジェクト作成後に他のプロジェクトからコピーするなどがあります。②は新規作成後に「プロジェクトにファイルを追加」を行う必要があります。

③については、HEW の「プロジェクト」メニューには「ファイルの追加」、「ファイルの削除」などのコマンドがありますが、実際にコピー、削除は行われず、プロジェクトの設定ファイルにファイルが登録、登録削除されるだけですので注意が必要です。

「Select RTOS」ウィンドウでは、リアルタイム OS の設定、プロジェクトに含むスタートアップファイルの登録(コピー) を行います。

- Target type：「R8C/Tiny」
- RTOS（リアルタイム OS）：「none」
- Startup file type：「User」
  プロジェクトに含まれるファイルを指定します。「Default」に設定すると M16C コンパイラ NC30WA で提供する標準のスタートアップファイル ncrt0.a30 とセクションファイル sect30.inc がプロジェクトフォルダにコピーされます。自分で独自に作成したファイルを選択する場合は「User」に設定し、「Add」ボタンを押してファイルエクスプローラでそれぞれのファイルを指定してプロジェクトフォルダにコピーします。ここでは「User」に設定してサンプルプログラム「LED_PICOPICO」で使用しているスタートアップファイル、セクションファイル、ヘッダファイル、C ソースファイルをプロジェクトフォルダにコピーすることにします。

設定が終わったら「Next」をクリックしてください。

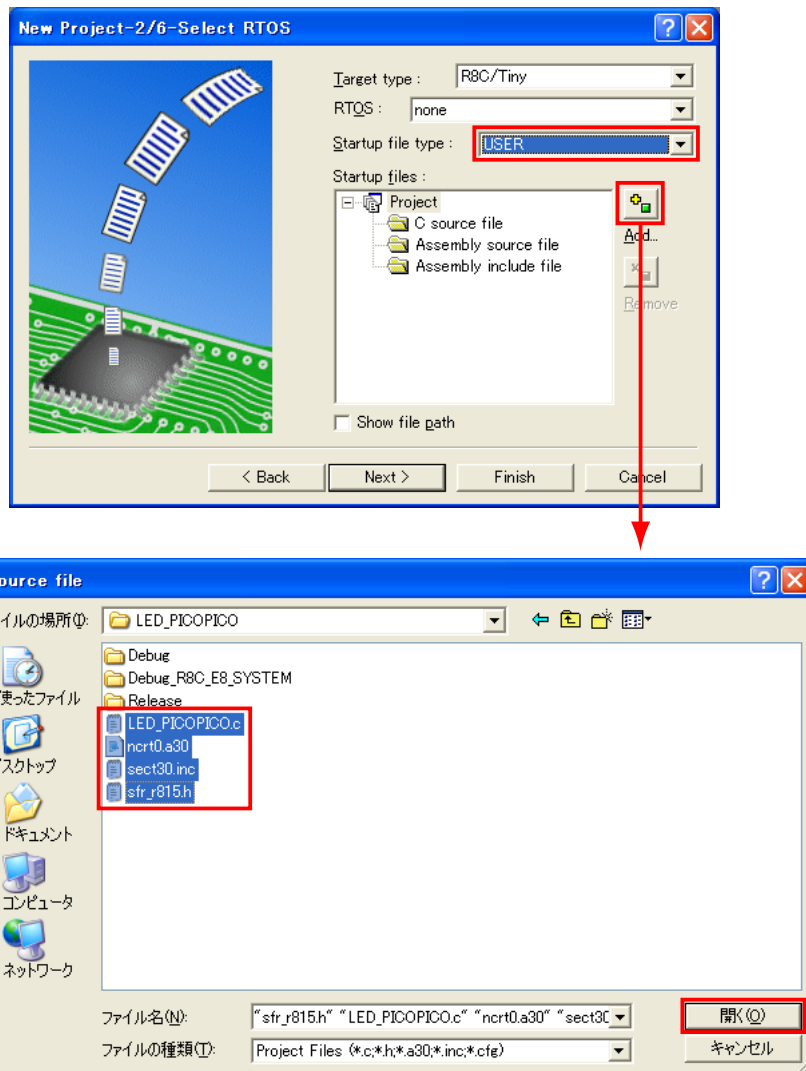



**図4-4 HEW: Select RTOS**

「Setting the Contents of Files to be Generated」ウィンドウでは標準I/O関数、ヒープ領域の設定を行いますが、先の「SelectRTOS」ウィンドウで「Startup file type」を「Default」に設定した場合のみ設定可能です。

- Generate main() Function：「None」
  main関数のファイルを作成するかどうかを指定します。「C source file」に設定すると、プロジェクトフォルダにプロジェクト名と同じCソースファイル(プロジェクト名.c)が作成されます。「None」に設定すると、Cソースファイルは作成されません。ここでは、Cソースファイルはサンプルプログラム「LED_PICOPICO」よりコピー済みなので、「None」に設定します。

設定が終わったら「Next」をクリックしてください。

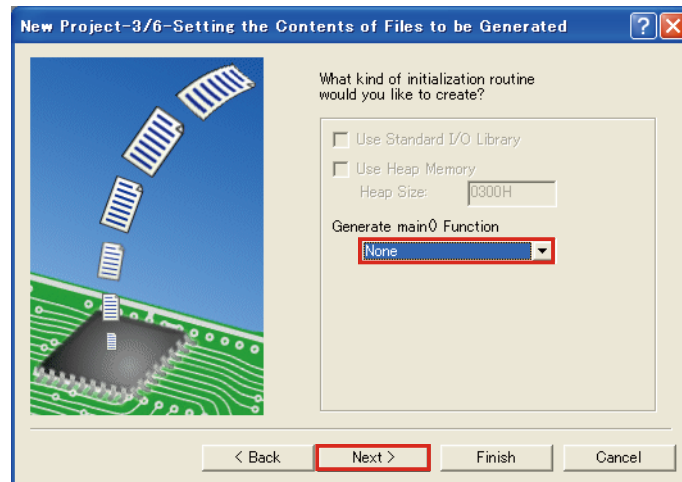



図4-5 HEW: Setting the Contents of Files to be Generated

「Setting the Stack Area」ウィンドウでは、ユーザースタックサイズと割り込みスタックサイズを設定しますが、「SelectRTOS」ウィンドウで、「Startup file type」を「Default」に設定した場合のみ設定可能です。そのまま「Next」をクリックしてください。

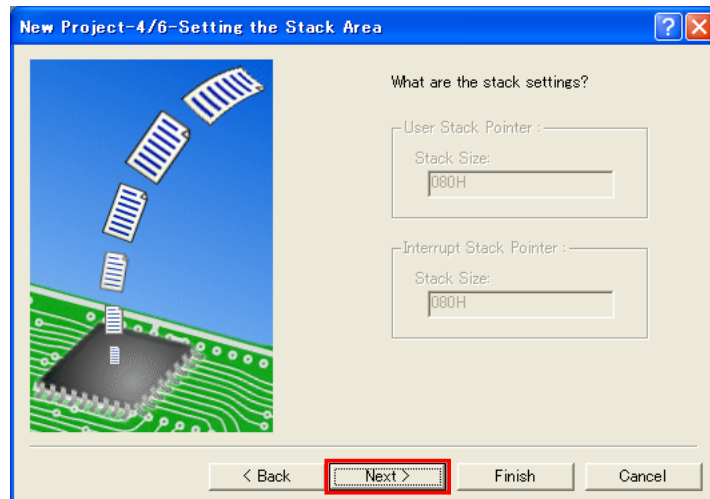



図4-6 HEW: Setting the Stack Area

「Target System for Debugging」ウィンドウでは、デバッグ環境を指定します。

- Targets：「R8C E8 SYSTEM」にチェックをいれる
- Target Type：「R8C/Tiny」

設定が終わったら「Next」をクリックしてください。

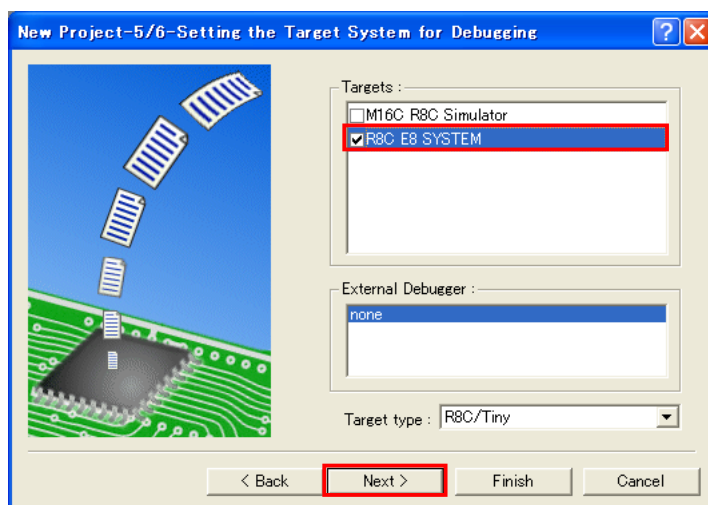



**図4-7 HEW: Target System for Debugging**

「Setting the Debugger Options」ウィンドウでは、特に設定することはありません。「Next」をクリックしてください。

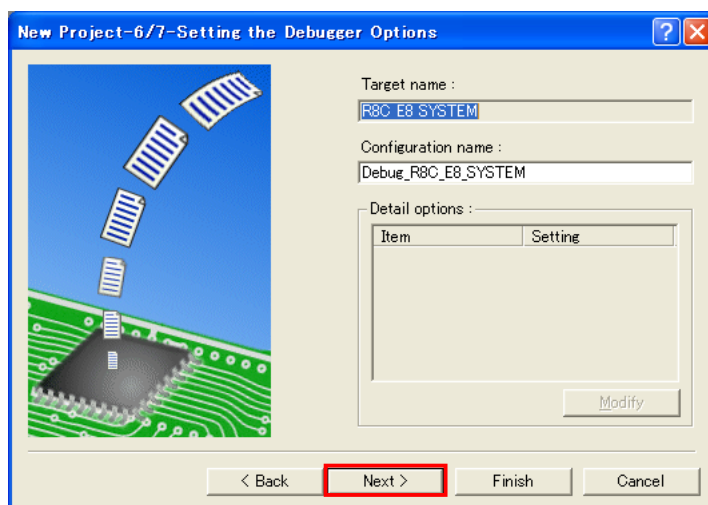



**図4-8 HEW: Setting the Debugger Options**

「Changing the File Names to be Created」ウィンドウでは、ファイル名を変更することができます。サンプルプログラム「LED_PICOPICO」よりコピーした C ソースファイル「LED_PICOPICO」を、プロジェクト名にあわせて「LED_PICOPICO2」と変更してください。「Next」をクリックしてください。

※ C ソースファイル名は､ビルドによって生成されるオブジェクトファイル名には反映されませんが（オブジェクトファイル名は、プロジェクト名が付けられます）、混同を防ぐため、他のプロジェクトから C ソースファイルをコピーした場合は、名前を変更することをお勧めします。

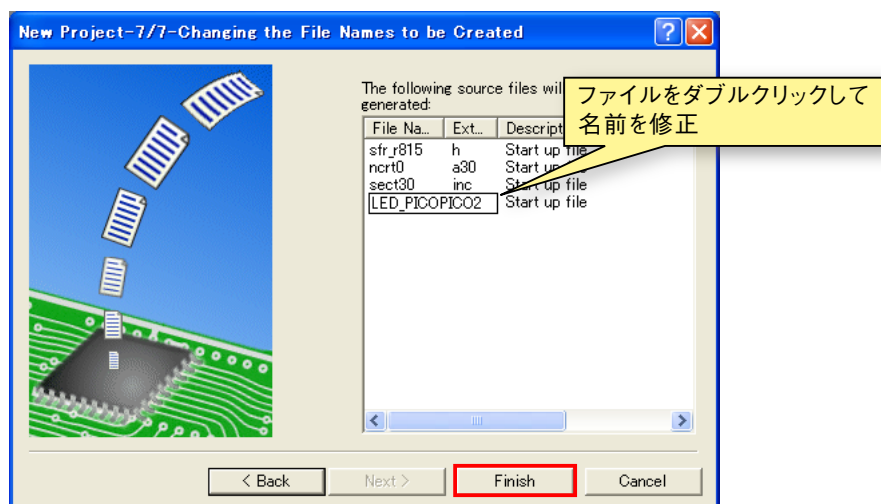



図4-9 HEW: Changing the File Names to be Created

「Summary」ウィンドウに設定した項目が表示されます。「Generate Readme.txt as a summary file in the project directory」にチェックを入れておくと、プロジェクトフォルダに「Summary」ウィンドウに表示されている各設定をテキストファイルにした Readme.txt ファイルを作成します。

「OK」をクリックするとワークスペース、プロジェクトの新規作成が終了します。

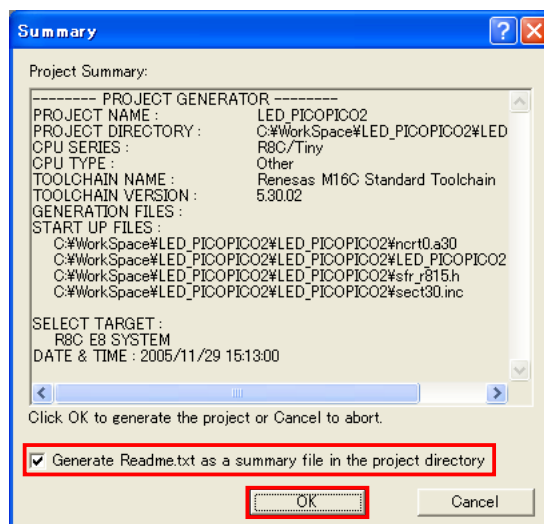



図4-10 HEW: Summary

## 4.3 ソースファイルの編集と保存

プロジェクトスペースの「LED_PICOPICO2.c」をダブルクリックして、エディタウィンドウを開き、コーディングします。コーディングが終了したらメニューバーの「ファイルの保存」アイコンをクリックするか、「Ctrl + S」キーを押してファイルを保存してください。

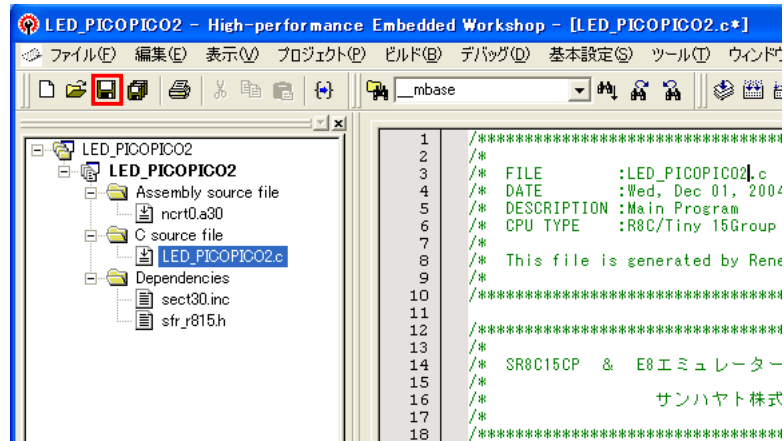



図 4-11 HEW:ファイルの保存

以下にサンプルプログラム「LED_PICOPICO2.c」、「sect30.inc」の参考リストを示します。タイマ X の割り込みで 0.1 秒間隔の LED 点滅を実現しています。赤枠内に「LED_PICOPICO」からの変更箇所を示します。

```
/*------------------------------------------------------------------------
 * FILE          :LED_PICOPICO2.c
 * DATE          :2006/3/20
 * DESCRIPTION   :Main Program
 * CPU TYPE      :R8C/Tiny 15Group (20MHz)
 * This file is generated by Renesas Project Generator(Ver.4.00.03.001)
 *------------------------------------------------------------------------*/


/*------------------------------------------------------------------------
 *
 * SR8C15CP　サンプルプログラム（タイマX割り込みによる0.5秒間隔LED点滅）
 *
 *                 サンハヤト株式会社
 *------------------------------------------------------------------------*/
/* インクルードファイル */
#include "sfr_r815.h"                   /* SR8C15CPのSFRレジス
/* プロトタイプ宣言 */
void init(void);                        /* システムクロック、SFR(Special Function Register)の初期設定*/
/*割り込み処理関数宣言*/
#pragma interrupt timer_x               /* ポートの状態を反転して出力する割り込み関数 */
/*------------------------------------------------------------------------
 * ID            : なし
 * 関数名        : main
 * 機能          : LEDを点滅させる
 * 引数          : なし
 * 戻り値        : なし
 * 使用関数      : init()
 *------------------------------------------------------------------------*/

void main(void)
{
        init();                         /* システムクロック、SFRの初期設定 */

        while(1)
        {
                ;                       /* 割り込み待ち */
        }
}
```



リスト 4-1 LED_PICOPICO2.c 各定義、メイン関数

```
/*------------------------------------------------------------------------
 * ID                 : なし
 * 関数名              : init
 * 機能                : システムクロックにメインクロック（外付けセラミック発振子20MHz）を設定、SFRの初期設定
 * 引数                : なし
 * 戻り値              : なし
 * 使用関数            : なし
*-------------------------------------------------------------------------*/
void init(void)
{
        /* システムクロック初期設定 */
        asm("FCLR I");                  /* 割り込み禁止 */

        prc0 = 1;                       /* CM0,CM1,OCDレジスタ プロテクト解除 */
        cm13 = 1;                       /* Xin Xout端子に設定 */
        cm15 = 1;                       /* Xin-Xout 駆動能力: HIGH */
        cm05 = 0;                       /* メインクロック発振 */
        cm16 = 0;                       /* CM16,CM7=00 →システムクロック分周なし */
        cm17 = 0;
        cm06 = 0;                       /* システムクロック分周比→CM16,CM17有効 */
        asm("nop");                     /* 発振安定までウェイト */
        asm("nop");
        asm("nop");
        asm("nop");
        ocd2 = 0;                       /* システムクロックにメインクロックを選択 */
        prc0 = 0;                       /* CM1,CM2,OCDレジスタ プロテクト設定 */

        asm("FSET I");                  /* 割り込み許可 */

        /* 入出力ポート初期設定 LED1=p1_2、LED2=p1_1 */
        drr2 = 1;                       /* p1駆動能力制御レジスタdrr2(LED1)に1を設定する */
        drr1 = 1;                       /* p1駆動能力制御レジスタdrr1(LED2)に1を設定する */
                                        /* 1に設定すると1ピンあたり約15mAの電流を引き込める */
        p1 = 0b00000010;                /* p1レジスタにLEDの初期状態（p1_2(LED1)→0:点灯、p1_1(LED2)→1:消灯）を設定する */
                                        /* この時点ではまだ入力ポートの設定になっているのでリードモディファイライト命令は使用しない */
        pd1_2 = 1;                      /* p1方向レジスタp1_2(LED1)を出力に設定する */
        pd1_1 = 1;                      /* p1方向レジスタp1_1(LED2)を出力に設定する */

        /* タイマX初期設定 */
        txic = 0b00000001;              /* タイマX割り込み制御レジスタ→割り込みレベル1 */
        txmr = 0b00000000;              /* タイマXモードレジスタ→タイマモード、カウント停止 */
        tcss = 0b00000001;              /* タイマXカウントリソース設定レジスタ→f8(20MHz/8=2.5MHz) */
        prex = 250-1;                   /* タイマXレジスタ、プリスケーラにカウント値設定 */
        tx = 200-1;                     /* 0.4μsec*50000=20msec　5000を250*200に分けて設定する */
        txs = 1;                        /* タイマXカウント開始 */
}
```

**リスト 4-2 LED_PICOPICO2.c 初期設定**

```
/*--------------------------------------------------------------------------
 * ID                  : なし
 * 関数名              : timer_x
 * 機能                : 2つのLEDを点滅させる（MB-RS8→p1_1：LED4、p1_2：LED5　CT-208→p1_1：LED2、p1_2：LED1）
 * 引数                : なし
 * 戻り値              : なし
 * 使用関数            : なし
 *--------------------------------------------------------------------------*/
void timer_x(void)
{
          static unsigned char timerX_count_500ms;              /* 500ms計測用カウンタ */
          timerX_count_500ms++;
          if( timerX_count_500ms >= 25 )                        /* 20msec×25=500msec */
          {
                    p1_2 = ~p1_2;                               /* p_2反転 */
                    p1_1 = ~p1_1;                               /* p1_1反転 */
                    timerX_count_500ms = 0;                     /* 500ms計測用カウンタ初期化 */
          }
}
```

割り込み処理関数では、割り込み回数をカウントし、25 回割り込みが発生した時に LED の状態を反転しています。

**リスト 4-3 LED_PICOPICO2.c タイマX割り込み関数**

```
;--------------------------------------------------------------
; variable vector section
;--------------------------------------------------------------
          .section   vector,ROMDATA   ; variable vector table
          .org       VECTOR_ADR

          .lword     dummy_int              ; vector 0
          .lword     dummy_int              ; vector 1
          .lword     dummy_int              ; vector 2
          .lword     dummy_int              ; vector 3
                          ― 省略 ―
          .lword     dummy_int              ; vector 17
          .lword     dummy_int              ; vector 18
          .lword     dummy_int              ; vector 19
          .lword     dummy_int              ; vector 20
          .lword     dummy_int              ; vector 21
          .glb       _timer_x
          .lword     _timer_x               ; vector 22
          .lword     dummy_int              ; vector 23
          .lword     dummy_int              ; vector 24
```

可変ベクタテーブルのタイマ X の該当するベクタに割り込み処理関数のアドレスを設定します。
セクション定義ファイル sect30.inc は、スタートアップルーチン ncrt0.a30 にインクルードされ、アセンブラが処理するので、C ラベル名の前にアンダースコア "_" を付ける必要があります。
また実際のラベルは LED_PICOPICO2.c ファイルにあるので、外部シンボル参照の指示命令（.glb）が必要です。

**リスト4-4 sect30.inc　可変ベクタテーブルへのタイマX割り込み関数アドレス設定**

## 4.4 ビルドオプションを設定する

「ビルド」メニューの「Renesas M16C Standard Toolchain」をクリックしてください。「Renesas M16C Standard Toolchain」ウィンドウが開き、ビルド時の各フェーズ（アセンブラ、コンパイラ、リンカ等）のオプションを設定することができます。

ここではLMC30（Load Module Converter）のIDコードオプションで、IDコードすべてにFFhを設定する例を説明します。この設定はリリースコンフィギュレーション「Release」に登録します。（コンフィギュレーションを「Release」にした状態でビルドしたときのみ、設定したオプションが有効になります。）

以下のように設定してください。

- 登録するコンフィギュレーション：「Release」
  対象となるプロジェクトは「All loaded Project」でも「LED_PICOPICO2」でもかまいません。
- 設定画面のタブ：「Lmc」（Load Module Converter）
- Category：「Code」
- [-ID] ID code check ID code setting：チェックを入れる
  IDコードを全てFFhに設定する場合は、パラメータを入れる必要はありません。

「Options Lmc :」エリアに現在設定されているオプションが表示されます。

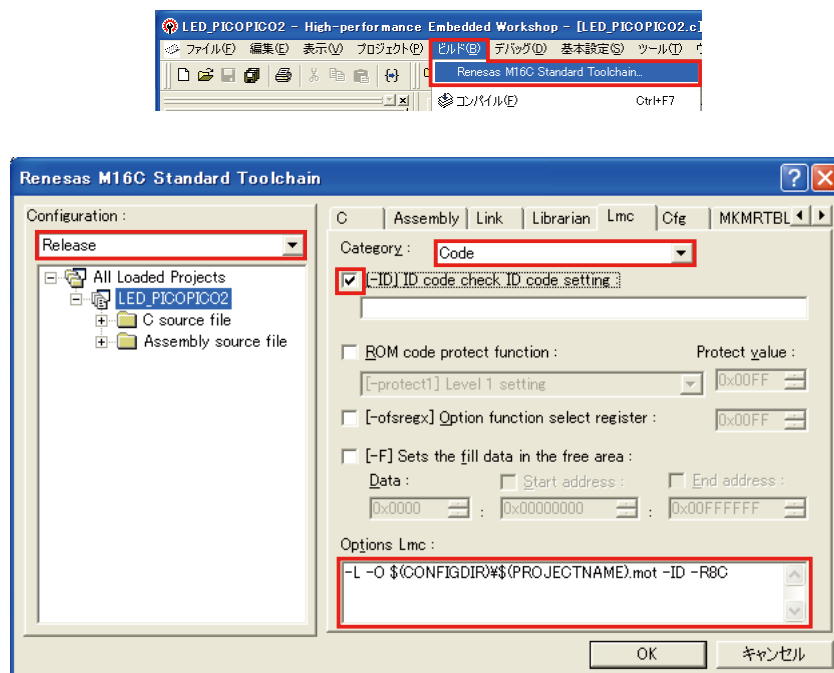



**図 4-12 HEW: -IDオプションの設定**

## 4.5 デバッグオプションを設定する

「デバッグ」オプションを必要に応じて設定しておきます。

「デバッグ」メニューの「デバッグの設定」をクリックし、設定画面タブを「オプション」にしてください。対象となるデバッグセッションは、「SessionR8C_E8_SYSTEM」に設定します。必要に応じて、各オプションにチェックを入れてください。

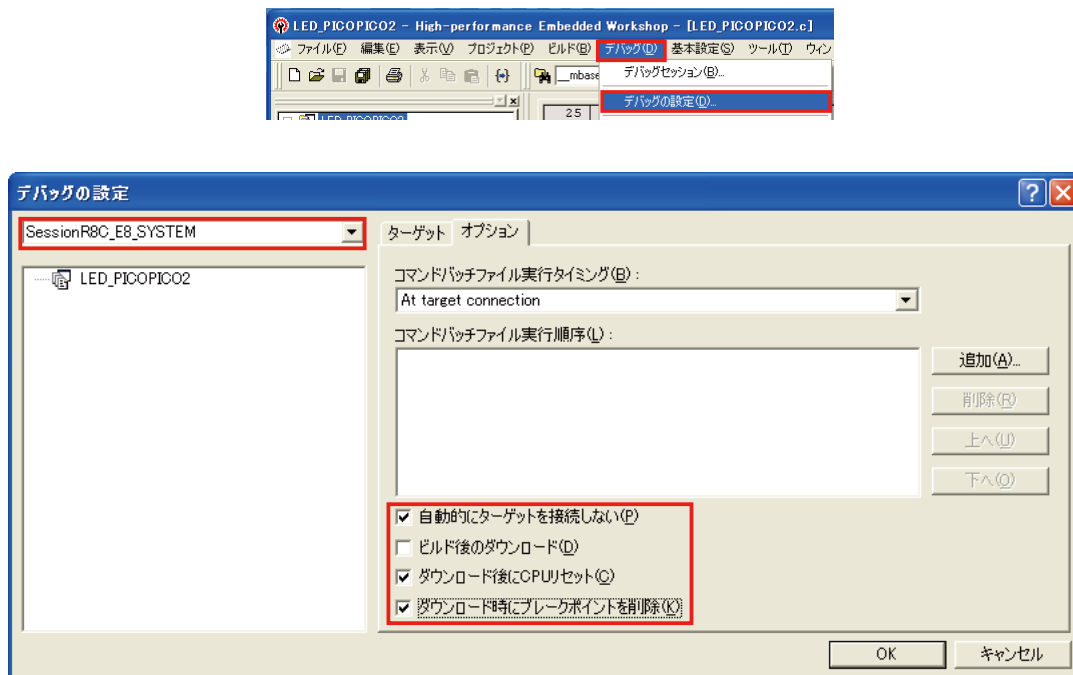



**図 4-13 HEW: デバッグオプションの設定例**

- **自動的にターゲットを接続しない**
  デバッグセッション「SessionR8C_E8_SYSTEM」に切り替えるたびに、またはデバッグセッションが「SessionR8C_E8_SYSTEM」の状態でコンフィギュレーションを切り替えるたびに E8 の接続ウィザードが開始されないようにします。このオプションを設定した状態で E8 と接続したい場合は、「デバッグ」メニューの「接続」をクリックするか、「ツール」メニューの「接続」アイコンをクリックします。

- **ビルド後のダウンロード**
  E8 と接続している状態でビルドを行うたびに、ダウンロードをするようにします。

- **ダウンロード後に CPU リセット**
  プログラムのダウンロードを行うたびに、ターゲットマイコンをリセットするようにします。

- **ダウンロード時にブレークポイントを削除**
  プログラムのダウンロードを行うたびに、前回設定したブレークポイントを削除するようにします。

ここまでの設定で、プロジェクトを一度保存してください。

あとは、「3. サンプルプログラムを動かそう」を参照して、ビルド、E8 エミュレータとの接続、プログラムのダウンロード / 実行を行ってみてください。

# 改訂履歴

| Rev. | 発行日 | ページ | 改訂内容 |
|---|---|---|---|
| 1.00 | 2005/05/01 | - | 初版発行 |
| 2.00 | 2006/04/10 | - | HEW3 から HEW4 へのバージョンアップに伴う修正 |

**R8C/Tiny マイコンシリーズ**
**スタートアップガイド　E8 エミュレータ導入編**

発行日　2006 年 4 月 10 日 Rev2.00

発　行　サンハヤト株式会社　〒 170-0005 東京都豊島区南大塚 3 丁目 40 番 1 号



SG043101